UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

FAKULTÄT FÜR
ELEKTROTECHNIK,
INFORMATIK UND
MATHEMATIK

Evolving Belief Network for Resource-efficient Place Recognition in Unknown Environments

Von der Fakultät für Elektrotechnik, Informatik und Mathematik
der Universität Paderborn

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation

von

MS-CSE Syed Muhammad Ali Musa Kazmi

Erster Gutachter:	Prof. Dr. Erdal Kayacan
Zweiter Gutachter:	Prof. Dr. Sybille Hellebrand

Tag der mündlichen Prüfung: 26.06.2025

Paderborn 2025

Diss. EIM-E/388

Dedication

Dedicated to my family who stood firm with high patience in tough times and their moral support enabled me to successfully complete the doctoral research.

Declaration

I hereby declare that I have completed the work of this PhD dissertation with my own efforts and no part of this work or documentation has been copied from any other source. It is also assured that this work is not submitted to any other institution for award of any degree or certificate.

Paderborn, August 17, 2025

MS-CSE Syed Muhammad Ali
Musa Kazmi

Abstract

Remembering and recalling places is an essential part of navigation in biological systems, e.g., humans. While these skills develop progressively in a subconscious mind, teaching technical systems the mastery of navigation is a challenging task. The complexity of the matter arises from the fact that the position of a vehicle can not be precisely estimated using motion sensors. A conventional way to tackle this problem is to use additional sensors, such as an optical camera, and track the geometric pose of the vehicle in a Cartesian map. The geometric notion of mapping places is contrary to how biophysical systems learn the mental map of environments using visual cues. Being inspired from this characteristic, many state-of-the-art technical systems perform mapping in the space of visual appearance to locate the vehicle. This is where place recognition emerges as a tool for appearance-based mapping. Despite ample research in place recognition, the mainstream methods rely on offline training or environment-related parameter tuning. Fulfilling such demands is impractical in unknown scenarios, as nothing is known beforehand about the environment. As a result, these systems undergo severe performance degradation when deployed in a previously unseen environment. In this research, we addressed the challenges of online visual learning and place recognition in unknown environments. The biological aspect of this work is to exploit the strengths of human vision and learning system. In this regard, we captured the human-like scene representation using a bank of Gabor filters and fused the nearby context to obtain the visual description of a place. To realize on-the-fly visual learning, an evolutionary version of the self-organizing neural network is proposed, which mimics the competitive behavior of the cells found in visual and perirhinal cortices. Given the incrementally trained network, the place recognition challenge is modeled using Bayesian statistics. Thus, each neuron in the evolving network possesses a belief of representing the query place. For the decision of loop-closure event, we aggregate the activation strength of a sequence of highly active neurons, which we called the expected network activity. The algorithm's performance is demonstrated on various challenging test sequences, which are recorded at different daytimes, under extreme lighting conditions and in presence of dynamic objects. Compared to the best baseline method (i.e. SeqSLAM), our algorithm delivers 15.2% higher place recognition accuracy and 42.5% improved runtime on the routes as large as $\sim$18 km.

Zusammenfassung

„Place Recognition" ist ein wesentlicher Bestandteil der Navigation in biologischen Systemen, z.B. bei Menschen. Hier entwickeln sich Fähigkeiten schrittweise im Unterbewusstsein, dadurch ist es eine herausfordernde Aufgabe, dieses Können einem technischen System beizubringen. Die Vielschichtigkeit der Aufgabe entsteht dadurch, dass Bewegungssensoren die Position eines Fahrzeugs nicht präzise schätzen können. Eine klassische Lösung dieses Problems ist, zusätzliche Sensoren, z.B. optische Kameras, einzusetzen und die Position des Fahrzeugs in einer geometrischen Karte zu verfolgen. Dagegen lernen biophysikalische Systeme Umgebungen ohne präzise geometrische Informationen mit Hilfe von visuellen Merkmalen. Inspiriert von diesem Eigenschaften führen viele moderne technische Systeme eine Kartierung im visuellen Bereich durch, um Fahrzeuge zu lokalisieren. Hier taucht die Szenenerkennung als einen Werkzeug für erscheinungsbasierte Kartenerstellung auf. Trotz umfangreicher Forschung auf dem Gebiet „Place Recognition" sind bestehende Methoden von Offline-Training oder umgebungsbezogener Parameteroptimierung abhängig. In der Tat ist es unmöglich o.g. Anforderungen zu erfüllen, weil vorher nichts über die Umgebung bekannt ist. Vom Grund daher erfahren solche Systeme einen starken Leistungsabfall in unbekannten Umgebungen. In dieser Arbeit werden Herausforderungen des Online-Lernens und der Erkennung der Orten in unbekannten Gebieten betrachtet. Der biologischer Aspekt in dieser Arbeit besteht darin, die Stärken des menschlichen Seh- und Lernsystems zu nutzen. In diesem Zusammenhang werden visuelle Repräsentationen der Szene mit unterschiedlichen Gabor-Filtern extrahiert und mit der Nachbarschaft der Szene vereinigt. Um einen Online-Lernalgorithmus zu realisieren, wird ein evolutionäres, selbstorganisierendes neuronales Netzwerk entwickelt, welches das Konkurrenzverhalten der Zellen im visuellen und perirhinalen Kortex nachahmt. Anschließend wird das Place Recognition-Problem mithilfe der Bayesschen Statistik modelliert. Um eine Entscheidung für den Schleifenschluss zu treffen, wird die Aktivierungsstärke einer Sequenz hochaktiver Neuronen aufsummiert. Die Leistungsfähigkeit des Algorithmus wird mit verschiedenen Standardtestsequenzen getestet. Im Vergleich zu besten Verfahren (i.e. SeqSLAM) erzielt unser Algorithmus 15,2% hohe Erkennungsgenauigkeit und 42,5% schnellere Ausführungsgeschwindigkeit auf Strecken mit einer Länger von bis zu ca. 18 km.

Acknowledgement

All the acclamation and appreciation are for Almighty Allah, the most Gracious and the most Merciful, and the Holy Prophet Muhammad (Peace Be Upon Him and His Family) whose gracious favor enabled me to complete this project successfully.

With humble and profound sense of devotion, I express my most sincere gratitudes to my respected supervisor Professor Dr.-Ing. Bärbel Mertsching whose scholarly guidance and encouragement has enabled me to overcome the hindrances that came in my work during the whole tenure of the doctoral project. It is her absolute support and professionalism that I have been able to effectively deal with challenges of the thesis, no matter what's the difficulty level.

I would extend my thanks to all my fellow colleagues for the friendly research atmosphere. Their constructive criticism and readiness for help has facilitated me to develop positively and kept me determined to achieve the project's goals.

Finally, I have no words to thank my worthy parents, who put their heart and soul to send me thousands of miles away from them for pursuing my dream of the higher education. Besides this, I pay heartfelt thanks to my siblings who through thick and thin stood firm and gave me moral support to focus on my research. Also, I appreciate the moral support and prayers of my wife who recently joined me and kept me motivated to achieve my goals – without them it would not have been possible.

Contents

1 Introduction

> *Imagination is more important than knowledge. For knowledge is limited, whereas imagination embraces the entire world, stimulating progress, giving birth to evolution... in scientific research (Einstein and Shaw 2012, pp. 49).*
>
> **— Albert Einstein**

Biophysical systems (including humans) navigate reliably through a diverse range of environments, while primarily relying on visual cues. Their mental map of the surroundings is so elastic that despite traveling long distances they are able to effectively locate and home themselves. Over the years, scientists have made ample efforts to teach technical systems, e.g., a mobile robot (or a vehicle), the mastery of navigation. To this end, a robotic system is typically equipped with the motion sensors and/or global positioning system (GPS).

The crux of the matter is that the motion of the robot cannot be precisely estimated due to hardware wear-out or other random factors, such as irregular or rugged terrain. Moreover, GPS does not work in all environments. For instance, in situations like densely vegetated areas, underground tunnels, and indoors, the GPS signal is either poor or remains disrupted. If the motion error is left unnoticed, it accumulates over time and thus yields distorted trajectory that significantly deviates from the actual path traveled by the robot. Hence, the aid of the auxiliary sensors, e.g., optical camera, is taken to reduce the magnitude of error in the robot's pose (i.e. position and heading direction). These sensors perceive the change in position of the objects (aka landmarks) with regard to the robot's motion. This entire process of learning the environment's map and locating the vehicle within the learned map is called as *robotic mapping* or *simultaneous localization and mapping (SLAM)*.

Conventionally, the robotic mapping task is addressed using filtering or pose-graph optimization methods. The difference between the two approaches is that the former tracks most recent pose of the vehicle over the consecutive movements to minimize the error in estimated trajectory. Whereas, the latter finds the most likely pose configuration of the vehicle out of all possible poses that may happen based on certain constraints. Regardless of the method used, ensuring the *global consistency*

of a map is a paramount concern for long-term navigation. A map is said to be globally consistent, if upon re-visiting the previously traversed route, the error in estimated vehicle's pose is within the acceptable bounds. That is to say, the vehicle must be able to detect the re-observation of the familiar landmarks (i.e. previously seen features), known as *loop-closure detection*.

From a traditional standpoint, the loop-closure detection is done in the coordinate space of the map. As a result, the recognition of familiar features remains highly susceptible to wrong matches. Moreover, finding correct associations in a massive collection of mapped features gets computationally intensive with the ever-increasing size of the explored environment. A recent comparative study, see Section 2.5.3, demonstrates that loop detection in the space of visual appearance is far more robust and efficient than finding loop closures in a geometric space. This is where the place recognition emerges as a tool for robust loop detection. Also, at this point researchers argue, if there is a potential need of building accurate geometric maps of highly dynamic environments, or learning topologically correct maps in the space of visual appearance, aka appearance-based mapping, suffices. Essentially, the appearance-based mapping is analogous to the way biological systems recognize places for navigating through environments, without learning the precise geometry of the physical space.

Despite ample research in place recognition, the trend that continues to exist across the mainstream methods is the need of offline training or environment-related parameter tuning prior to the system's deployment. As a result, these methods undergo severe performance degradation, when applied to previously unobserved environments. Hence, place recognition in unknown environments is generally the focus of this thesis. This makes the optical camera an obvious choice in our research. Besides biological motives, cameras are cheap, easily available, and above all offer rich features to effectively deal with the cluttered environments.

1.1 Motivations

There are several motivations underpinning this research work.

1.1.1 Biological Motives

One of the key motivations of this research is to study how biophysical systems per–ceive surroundings, learn the stimulus and exploit the visual information to navigate through heterogeneous environments. This would serve as a basis to improve the place recognition performance for large-scale appearance-based mapping. As the

visual perception and navigation are dealt on the fly in a mammalian's brain, online visual learning and place recognition are at the top priority in this thesis. Moreover, it is the elasticity of the mental map that the massive collection of places is stored compactly. Hence, this work is motivated to compactly learn the topologically correct representation of the environment using only visual information.

1.1.2 Scientific Aspect

Among the mainstream place recognition approaches, the drawback that repeatedly appears is their inability to deal with previously unseen environments. Indeed, this downside arise from the fact that the majority of the existing systems need offline training, assume constant velocity along the routes, or require environment-related parameter tuning. Meeting these absolute necessities is impractical, when the system has to applied in an anonymous environment under uncontrolled conditions. This motivates this research to perform place recognition in unknown environments, without the need of parameter tuning or environmental-related offline training.

Appearance-based mapping does not limit the usefulness of our work to a non-metric system. This work can also be integrated with the filtering or pose-graph optimization methods for adding the metric awareness to the mapping process.

1.1.3 Application Standpoint

A robust solution to place recognition facilitates planning the maneuvers through environments (i.e. path planing) for goal-directed navigation. Ultimately, it will assist in many daily life applications, such as self-driving cars, service robots, cleaning bots, agricultural robots, assisting visually impaired people for navigation and others.

1.2 Research Questions

Before describing the research questions, it is here important to clarify whether a place and a location are different concepts and how a place is defined in this thesis? For details, we refer to Section 2.5.1. A place is more of a human's conceptualization of a location based on its characteristics, whereas a location has a geometric property. It is a unique coordinate pair in a Cartesian space that refers to a particular position. When it comes to defining a place, there is no agreed upon definition. For instance, in a global map, different subcontinents could be treated as places, whereas further down the hierarchy different buildings or tourist landmarks can be defined as a place. In this thesis, we use location and place interchangeably to refer to the same

concept. The rational for this simplification is that we do not use geometric data. Hence, each image is essentially a potential location (or a place) in this work.

The *first and the foremost* question that we address in this research is that of place recognition in unknown environments, without offline training or environment-re–lated parameter adjustment. Indeed, place recognition in unknown environments is a challenging task, as nothing is known beforehand about the environment and the conditions along the routes are unpredictable and volatile. For instance, dynamic objects may conceal the perceptual layout of scenes, or places may look visually different due to extreme lighting or shadows of the building or trees along the routes. The data arrives incrementally and keeping the data in physical memory over the longer runs is infeasible. Moreover, the limited field of view (FOV) of the camera aggravate the complexity of the place recognition task, such as viewpoint issues.

Secondly, to address the aforementioned objective, we take inspirations from biology and exploit the strengths of the human vision and learning system (i.e. certain areas of the visual cortex and the hippocampus) so that the natural responses to images can be extracted and learned incrementally. The psychological studies suggest that humans perceive the meaning a scene within a small fraction of a second, referred to as a *gist of the scene* (Goldstein 2014, pp. 109). Hence, we aim to find faster ways of obtaining the human-like layout of the scenes from images, such as (Oliva and Torralba 2001), instead of following complicated procedures like image segmentation or fine-grained image analysis.

The extracted scene description is not self-sufficient to model the representation of the environment. Hence, the *third question* is to study how the feature maps are learned in the early vision layers of the visual cortex and the associated areas of hippocampus. This would assist us to compactly learn the general representation of the environment, rather than storing every single image in a database. The type of neural network that mimics the competitive behavior of cells in cortical regions of the brain is known as self-organizing map (SOM). Hence, the feature maps learned by a SOM resemble the maps of features learned in cortical regions, follow details in Section 2.7.2. Unfortunately, a SOM is a fixed sized network and determining the appropriate number of neurons for a particular environment is infeasible, esp. when no information about the environment is available. Besides this, fixing the network's size will constrain its ability to learn new structures in the feature space. A rescue from this problem is an evolutionary version of the SOM, namely the Growing SOM (GSOM). However, unlike other machine learning algorithms, the design of the GSOM must be defined according to the task at hand. So, the goal is to propose novel learning dynamics for GSOM to realize online visual learning.

The *fourth aspect* is to describe a robust online place recognition system. As a Bayesian network is a robust tool for decision making and classification, we aim to apply Bayesian statistics to the evolving GSOM network and solve the place recognition challenge. In this manner, we perform simultaneously place learning and recognition. Altogether the system is regarded as a growing belief network (GBN), as each neuron in the network posses the belief (i.e. probability) with which it represents the query place. This information is later leveraged for loop detection.

Finally, the natural question that arises is the influence of the size of the environment on the scalability of the algorithm and its resource efficiency. Hence, for long-term place recognition, the goal is to deliver the real-time performance, while consuming as less memory as possible.

1.3 Contributions of this Research

The contributions of the research under-consideration could be discussed from the scientific and performance standpoints. The scientific aspect is that we have introduced a novel framework that simultaneously performs place learning and recognition for online appearance-based mapping. Performance-wise, this research introduces a solution that does not need environment-related tuning of parameters. On the routes as large as ca. 18 km, it delivers high-precision and resource-efficient loop detection performance compared to popular place recognition methods.

1.3.1 Scientific Perspective

Scientifically, this research contributes in three major aspects. The first key con–tribution is a biologically inspired *online visual learning* algorithm. The second contribution that builds on the top of it is a *probabilistic framework* for real-time place recognition. Lastly, we *enhance the visual description* of a scene through the visual appearance of the nearby locations.

The groundwork for *online visual learning* is laid in Chapter 4. To this end, we obtained the human-like perceptual layout of the scene through gist features. Next, we proposed the novel learning dynamics for GSOM to achieve on-the-fly visual learning. The algorithm learns the topological maps of the feature space such that the places sharing the similar perceptual structure exist close to one another. The algorithm is tested on the benchmark dataset in the presence of noise and motion blur in the images. Finally, to perform the weakly-metric mapping, the algorithm is integrated as a place recognition front-end to the RatSLAM system (Milford and Wyeth 2008). Thereon, a series of successive improvements are made in the

algorithm that appear in Chapters 5 and 6, respectively. These upgrades can be defined in terms of network initialization, learning rule, learning decay schemes, neural connectivity, network's growth control scheme, and neurons' instantiation method.

While in Chapter 4 the place recognition is achieved using nearest neighbor search (NSS), the first significant contribution for *real-time place recognition* appears in Chapter 5. In this regard, we utilized the learned network dynamics and computed the maximum a posteriori estimate via Bayesian statistics. Hence, the neuron that is maximally active to represent the query place is used for the decision of place recognition. Further improvements are proposed in place recognition algorithm in Chapter 6. To this end, we look out for the activity of sequence of maximizer neurons. If more than one neurons are consecutively active, it strengthens the belief that the vehicle is traversing a familiar route. Thus, we aggregate the activation strength of a series of maximizer neurons to decide on loop and no-loop locations.

Being able to uniquely describe the appearance of a scene is also essential for place recognition in dynamic and large-scale environments. To compose such a *robust scene description*, the nearby context of a location is added to the current location's description via exponential smoothing, see Chapter 6. Hence, the visual description of a location is the weighted average of the visual description of the neighboring locations. In this process, the nearest locations get more weight to form the description of the current location than the those that are farther away. Whereby, the spread of the neighborhood around a location is set based on frame rate.

1.3.2 Performance Perspective

The outcome of the aforesaid contributions can be articulated in terms of per–formance measures, such as accuracy, general applicability, system's scalablity to unknown and diverse environments and resource efficiency (in space and time).

Place Recognition Accuracy

Place recognition accuracy determines the combined performance of the system, when we put together all the contributions. To evaluate the system's performance, the precision–recall measure is utilized and an exhaustive set of experiments are performed on eleven challenging sequences. We then compared the results with the popular baseline algorithms, see Table 6.3. Compared to the best baseline method, our system shows 15.2% higher accuracy. Section 7.3 outlines the achievements of this research from quantitative and qualitative perspective.

General Applicability

A system is said to be generally applicable, if it tends to keep high-performance place recognition across the unknown environments, without having the need for environment-related training or parameter tuning. To this end, we obtained the standard test sequences from various environments. These sequences come up with different frame rates, route lengths, and image resolutions. Moreover, some of the sequences are traveled at different daytimes and months apart. The test bench also includes the sequences with many loop closing routes, of which some are even traveled multiple times. Despite such a large variability and environmental differences between the datasets, the proposed system has delivered high-precision results for place recognition, without environment-specific parameter adjustment.

Scalability vs. Resource Efficiency

Scalability is the ability of an algorithm to gracefully adapt itself to the size and dynamic nature of the environments. In the context of online place learning and recognition, one of the important task is to handle large-scale environments. The typical behavior of the existing algorithms is that the more they get exposed to new parts of the environment, the harder it becomes to sustain high-precision performance due to perceptual aliasing. Apart from that the increasing size of the environment uplifts the demands for computational resources, such as processing power. For instance, convolutional neural networks (Sünderhauf et al. 2015b), occupy huge computational resources for large-scale place recognition. On the other hand, the approaches, such as sequence-based methods (Milford and Wyeth 2012), perform fast on small-size environments. However, their needs for processing power escalate rapidly as the size of the environment gets larger. In this research, our system has successfully demonstrated online place learning and recognition on the routes up to ca. 18 km long in the St. Lucia suburb (cf. Section 6.5.1). The sequence, e.g., SL08 (cf. Table 6.1), comprises 21,815 images, whereas our system instantiated just 2787 neurons (cf. Table 6.5) to learn the entire environment and achieved 42.5% faster execution time than the best baseline approach.

1.4 Thesis Structure

The rest of the thesis is organized as follows. Chapter 2 starts with the probabilistic definition of the SLAM problem. Thereon, we briefly describe the essential compo–nents of the vision-based SLAM, followed by the discussion on classical methods and the rise of place recognition techniques for robotic mapping. Finally, the chapter

introduces the reader to the concepts of visual perception, learning and navigation in biophysical systems, in the light of current biological findings.

Chapter 3 provides an in-depth review of the state-of-the-art developments in place recognition for appearance-based mapping. At the end, it tabulates the qualitative comparison of the existing approaches, with focus on their visual learning and recognition methodologies.

In Chapter 4, we introduce a novel bio-inspired model for on-the-fly visual learning. At first, the learning dynamics for the GSOM are defined. Thereafter, we investigate the place recognition module of the RatSLAM system and compare its performance with the proposed GSOM-based place recognition front-end (i.e. GSF). The robust–ness of the systems is evaluated on the benchmark dataset in the presence of noisy and blurry inputs. Eventually, mapping is performed by replacing the existing place recognition front-end with the GSF.

Chapter 5 proposes several changes in the learning dynamics of the GSF to address the challenges of place recognition in large-scale environments. Moreover, this chapter introduces a novel place recognition algorithm, which has its foundations in Bayesian statistics. This results in a standalone system that we named growing belief network (GBN). The system performs simultaneous place learning and recognition for appearance-based mapping. The performance of the algorithms is evaluated on seven challenging sequences from three different datasets.

Chapter 6 introduces the importance of nearby context for place recognition. At first, the reader is introduced to the improvements in the learning and recognition algorithms of GBN. The new system is thus named as (IGBN). The performance of the IGBN is evaluated on eleven challenging test sequences from the four different datasets. Moreover, the improvements are compared with GBN and the popular place recognition algorithms, including their timing efficiency.

Finally, Chapter 7 summarizes this thesis, outlines the progressive improvements in learning and recognition algorithms, and discusses the standing of this research work in relation to the state-of-the-art methods. Lastly, the further directions of improvements are anticipated.

2 Background

This chapter describes the essentials of the robotic mapping process. Initially, it introduces the reader to the mathematical definition of the robot localization and mapping. Thereon, the discussion extends to the vision-based map estimation, followed by the rise of place recognition in the domain of robotic mapping. Finally, we discuss the localization and mapping in biophysical systems, which is the one of the motivations of this work.

2.1 Simultaneous Localization & Mapping (SLAM)

A mobile robot navigating in an unfamiliar environment typically relies on motion sensors (e.g. wheel encoders) to track its position. However, these devices are prone to errors due to manufacturing inaccuracy, hardware wear out or other random factors, such as slippery, rugged or uneven terrain. As the robot moves, the errors in estimated robot position accumulate, making it infeasible to build a coherent map of the environment. To reduce the drifts in the map, auxiliary sensors, e.g., optical camera or laser range finders, are employed, and the robot's motion is corrected by observing the static objects in the surroundings. Such a process of estimating the position of a vehicle and concurrently learning a spatial representation of the observed world (a map) is referred to as *Simultaneous Localization and Mapping* (SLAM) or *Robotic Mapping*.

Probabilistic techniques are of particular interest for robotic mapping due to their robustness to deal with the motion errors and noisy sensor data (Thrun et al. 2005). Almost every probabilistic method today is grounded in *Bayesian statistics*. To highlight the significance of Bayesian estimation in robotic mapping, let us consider a mobile robot operating on a flat ground, as depicted in Figure 2.1. Say, at a discrete time k, the robot has a pose configuration $\mathbf{x}_k = (x, y, \theta)$, where (x, y) is the position coordinate pair and θ is the heading direction. All the pose configurations acquired by the robot up to present time t are given as:

$$\mathcal{X}_t = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, \mathbf{x}_k, \mathbf{x}_{k+1}, \ldots, \mathbf{x}_t\}$$

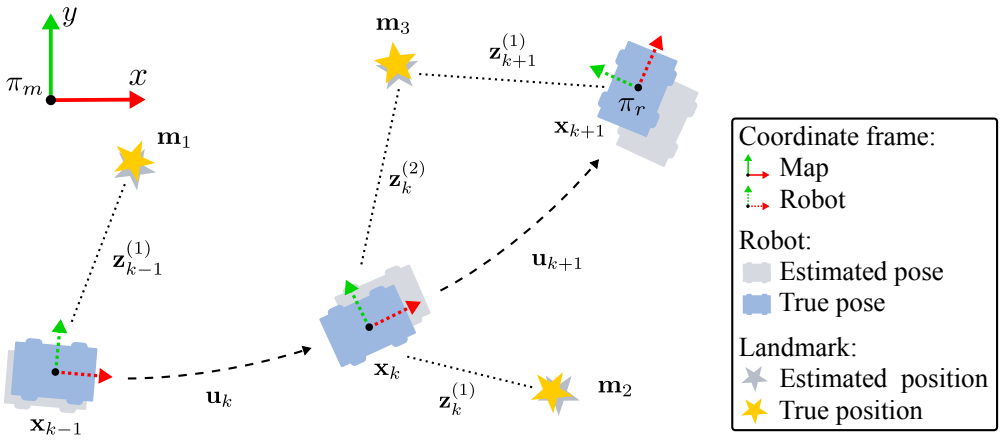


Figure 2.1: An arbitrary pose configuration of a mobile robot with the respective landmarks' observations in a two-dimensional space. The robot pose and the landmarks' positions cannot be precisely estimated due to motion errors and sensor noise.

The robot's movement between the two successive poses $\mathbf{x}_{k-1}$ and $\mathbf{x}_k$ is given by the motion command $\mathbf{u}_k$, a 2-tuple vector (v, ω) representing the linear and angular velocities, respectively. Hence, the sequence

$$\mathcal{U}_t = \{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k, \ldots, \mathbf{u}_t\}$$

describes the robot's motion between the respective poses. As the motion is prone to errors, the landmark are observed with the aid of auxiliary sensors:

$$\mathcal{Z}_{1:t} = \{\mathcal{Z}_1, \mathcal{Z}_2, \ldots, \mathcal{Z}_k, \ldots, \mathcal{Z}_t\}$$

Note that $\mathcal{Z}_k = \left\{ \mathbf{z}_k^{(i)} \in \mathbb{R}^d \,\middle|\, i = 1, \ldots, N_l \right\}$ is the set of landmarks sensed at the pose configuration $\mathbf{x}_k$, where each landmark observation $\mathbf{z}_k^{(i)}$ is a d-dimensional real vector and N_l is the number of observed landmarks. The landmarks that are static or detected in successive robot movements are saved in the map of the environment $\mathcal{M} = \{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_M\}$ relative to the global pose of the robot. Here, M is the total number of point landmarks in the map. Typically, a mapped landmark $\mathbf{m}_j : \forall j \in [1, M]$ is encoded as a two- or three-dimensional vector. However, in practice it is of higher dimensions comprising the point description of the landmarks.

2.1.1 Prediction of the Robot Pose

Evolution of a pose from time $t-1$ to the present time t can be described as:

$$p_{\mathcal{X}}(\mathbf{x}_t) \approx \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}, \mathcal{U}_t)\, p_{\mathcal{X}|\mathcal{Z}}(\mathbf{x}_{t-1}) \tag{2.1}$$

The transition probability $p(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}, \mathcal{U}_t)$, also known as *motion model*, relates the robot poses $\mathbf{x}_{t-1}$ and $\mathbf{x}_t$ as a function of motion commands. According to *Markov*

assumption, future state of a system can be completely described by the present state and the associated control commands. Hence, the knowledge of the previous control commands $\mathbf{u}_{1:t-1}$ does not directly influence the future pose $\mathbf{x}_t$. We therefore re-express the Eq. (2.1) as:

$$p_{\mathcal{X}}(\mathbf{x}_t) \approx \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_t)\, p_{\mathcal{X}|\mathcal{Z}}(\mathbf{x}_{t-1}) \tag{2.2}$$

where the posterior distribution $p_{\mathcal{X}|\mathcal{Z}}$ is the belief of the robot in previous pose configuration $\mathbf{x}_{t-1}$; we discuss this probability distribution in the following section.

2.1.2 Correction of the Estimated Pose

The motion alone is insufficient to precisely estimate the robot's pose, hence the landmarks are observed and associated iteratively over the subsequent movements. This leads us to condition the present pose on sensor observations via Bayes rule:

$$p_{\mathcal{X}|\mathcal{Z}}(\mathbf{x}_t \mid \mathcal{Z}_{1:t}, \mathcal{U}_t) \approx \frac{p(\mathcal{Z}_t \mid \mathbf{x}_t, \mathcal{Z}_{1:t-1}, \mathcal{U}_t)\, p_{\mathcal{X}}(\mathbf{x}_t)}{p(\mathcal{Z}_t)} \tag{2.3}$$

Given the new pose configuration $\mathbf{x}_t$, the motion commands $\mathcal{U}_t$ and the previous observations $\mathcal{Z}_{1:t-1}$ provide no additional information of the new observations $\mathcal{Z}_t$ (conditional independence). The denominator term $p(\mathcal{Z}_t) = \sum_{\mathbf{x}_t} p(\mathcal{Z}_t \mid \mathbf{x}_t)\, p_{\mathcal{X}}(\mathbf{x}_t)$ is the marginal probability with respect to the state space. It assures that the probabili–ties over the state space sum to 1, and so it can be defined as a normalizing constant $\eta^{-1} = p(\mathcal{Z}_t)$. This simplifies Eq. (2.3) to the following expression:

$$p_{\mathcal{X}|\mathcal{Z}}(\mathbf{x}_t \mid \mathcal{Z}_{1:t}, \mathcal{U}_t) \approx \eta\, p(\mathcal{Z}_t \mid \mathbf{x}_t)\, p_{\mathcal{X}}(\mathbf{x}_t) \tag{2.4}$$

The likelihood term $p(\mathcal{Z}_t \mid \mathbf{x}_t)$ is a *measurement model*. It relates the robot-centric landmark observations to the current pose configuration and other landmarks in the map. The posterior $p_{\mathcal{X}|\mathcal{Z}}(\mathbf{x}_t \mid \mathcal{Z}_{1:t}, \mathcal{U}_t)$ is the belief in the new pose configuration after incorporating the measurements $\mathcal{Z}_t$. Substituting Eq. (2.2) in Eq. (2.4) gives a recursive Bayesian definition of the *robot localization* process:

$$p_{\mathcal{X}|\mathcal{Z}}(\mathbf{x}_t \mid \mathcal{Z}_{1:t}, \mathcal{U}_t) \approx \eta\, p(\mathcal{Z}_t \mid \mathbf{x}_t) \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_t)\, p_{\mathcal{X}|\mathcal{Z}}(\mathbf{x}_{t-1}) \tag{2.5}$$

Here, $p_{\mathcal{X}|\mathcal{Z}}(\mathbf{x}_{t-1})$ is the belief in the preceding pose configuration. Putting it another way, it is the posterior distribution at the state $\mathbf{x}_{t-1}$ after taking into account the measurements $\mathcal{Z}_{t-1}$, and yet before applying the motion command $\mathbf{u}_t$. So, the robot localization algorithm iterates between the two steps: robot moves and predicts the next pose using Eq. (2.2); after the motion, it observes the environment relative to the new pose and corrects the prediction via Eq. (2.4). The resulting posterior

distribution becomes the prior distribution for the prediction of the next pose. Note that Eq. (2.5) only accounts for the robot localization aspect. For a mapping task, one must also track the mapped landmarks' positions. To this end, the joint posterior distribution over the current state $\mathbf{x}_t$ and the map $\mathcal{M}$ is calculated, which can be obtained by re-expressing Eq. (2.5) as follows:

$$p_{\mathcal{X}|\mathcal{Z}}(\mathbf{x}_t, \mathcal{M} \mid \mathcal{Z}_{1:t}, \mathcal{U}_t) \approx \eta\, p(\mathcal{Z}_t \mid \mathbf{x}_t, \mathcal{M}) \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_t)\, p_{\mathcal{X}|\mathcal{Z}}(\mathbf{x}_{t-1}, \mathcal{M} \mid \mathcal{Z}_{1:t-1}, \mathbf{u}_{1:t-1}) \quad (2.6)$$

The above equation is generally regarded as a definition of the *online SLAM*, as it iteratively calculates the joint posterior over the most recent pose configuration $\mathbf{x}_t$ and the map $\mathcal{M}$ of the environment[1]. Figure 2.2 summarizes the sequential execution of the prediction and correction steps. This definition differs from the *full SLAM* problem, in which case one computes the joint posterior over the entire robot trajectory $\mathcal{X}_t$ and the map $\mathcal{M}$.

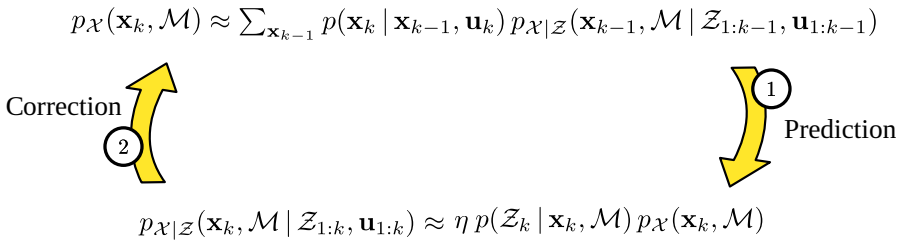


Figure 2.2: The SLAM algorithm incrementally iterates between the prediction and correction steps. The prediction step infers the robot's pose and the position of the landmarks in the map. In the correction step, also regarded as a *measurement update*, the observations $\mathcal{Z}_t$ are utilized to correct errors in the predictions.

[1] Note that the map variable ($\mathcal{M}$) does not depend on time, as the landmark locations in the map is assumed to be static. In practice, one has to establish correct correspondences between landmarks (cf. Section 2.2.2) for a correct map estimation.

2.2 Essentials of Vision-based SLAM

In recent years, optical cameras have been massively employed for map estimation. Their reasonable cost and easy availability for consumer applications make them a popular choice compared to range sensors, i.e., laser range finder and sonar. Additionally, camera imagery offers rich features to deal with the complexity of environments resulting from scale and appearance changes (Williams et al. 2009). It is also worth considering vision sensors because biological systems primarily rely on visual cues while navigating through diverse environments (Wyeth and Milford 2009). The succeeding discussion in the scope of this work focuses on vision-based SLAM techniques, unless otherwise stated. Mokssit et al. (2023) summarize recently developed deep learning methods to address a visual SLAM problem. Figure 2.3 highlights the key aspects of the vision-based SLAM.

The imagery acquired from a camera is often pre-processed, which generally includes cropping, down-sampling and de-noising (suppressing unwanted artifacts). Thereon, the front-end tasks, such as feature extraction, feature tracking and loop detection, are carried out and the information is fed to the map estimation algorithm(s). The path integration module calculates the global position of the robot based on motion commands and other inertial measurements. To correct the position errors that accumulate over subsequent movements, the map estimation algorithms rely on the feedback of the vision sensors. While the front-end operations: feature extraction, feature tracking, and loop-closure detection have been discussed in Sections 2.2.1 – 2.2.4, respectively, a brief overview of various map estimation techniques is presented in Section 2.4.

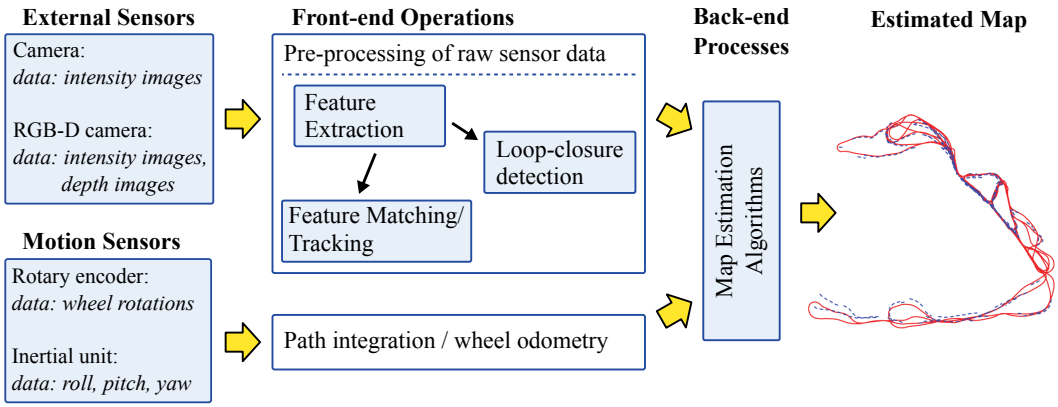


Figure 2.3: Modular description of the vision-based SLAM: Front-end modules acquire data from the motion and external sensors. The wheel-based odometry is oftentimes ignored, if the motion information is determined from the camera images. A map estimation algorithm yields the robot trajectory (red solid line), comparable to the true trajectory (blue dashed line), and the map of the environment.

2.2.1 Feature Extraction

A feature, also known as an *interest point* or a *keypoint*, is a pixel position in an image where the intensity change (gradient) is sharp in a local neighborhood of the pixel. As an image is a projection of a three-dimensional scene onto a two-dimensional plane, so in that respect the gradients in $x-$ and $y-$directions provide useful information for detecting features. The detected features are generally regarded as edges, corners or blobs. Typically, an image is slightly smoothen to filter out the noisy (or spurious) pixels using neighborhood information. For distinguishing the features from one another, the description in a local neighborhood of the pixel is stored in a vector as a numerical fingerprint, known as a *feature descriptor* or more concretely a *local feature descriptor*.

In last two decades, a variety of local descriptors have been introduced with applications in content-based image retrieval, image registration, object recognition and tracking. Nevertheless, only few of the existing descriptors have shown robustness to the brightness and geometric changes (e.g. scale and orientation), which are the paramount concerns to establish the correct feature correspondences across different images. The features that can be observed in multiple successive frames are indeed the good features to track (Shi and Tomasi 1994). Some popular feature descriptors include: SIFT (Lowe 2004), SURF (Bay et al. 2008), BRIEF (Calonder et al. 2010) and ORB (Rublee et al. 2011). A comparison of these and other descriptors is presented in (Carlevaris-Bianco and Eustice 2014; Krajník et al. 2017; Valgren and Lilienthal 2010).

2.2.2 Feature Matching

Finding the correct feature correspondences plays a significant role to calculate the self-motion of a vehicle (or a camera), referred to as *visual odometry* (VO). Moreover, it is a handy tool to recover the depth information, i.e., 3D positions of the feature points, from images. Feature matching techniques (Nistér et al. 2006; Scaramuzza and Fraundorfer 2011) detect a set of feature points separately in multiple (at least two) frames. The descriptors of the detected feature points are extracted and matched based on an appropriate distance metric, e.g., Euclidean, Hamming, etc. Finally, the relative camera motion between the frames is estimated using the best feature correspondences. As wrongly associated features lead to an inaccurate motion estimation, the outliers are rejected by applying the RANSAC (Random Sample Consensus) algorithm (Fischler and Bolles 1981), which iteratively searches for the solution with the minimum re-projection error.

2.2.3 Feature Tracking

Another paradigm that is particularly famous for estimating the self-motion of a camera is the feature tracking. The feature tracking has its fundamentals in optical flow estimation. The optical flow methods are correspondence-free (Fraundorfer and Scaramuzza 2012). They track either all the pixels (dense optical flow) or a sparse set of feature points (sparse optical flow) in adjacent frames. Given the assumption that the motion between the frames is very small and the brightness does not change, these approaches locally search the matching regions in adjacent frames around the interest points, instead of matching all the feature descriptors in the candidate images. For this purpose, usually normalized cross-correlation (NCC) or sum of squared differences (SSD) metric is used. In this manner, the information from the vector field can be utilized to estimate the camera motion, follow further discussion in (Yousif et al. 2015). However, the optical flow methods are useful only when the images are captured at a closer physical distance. If the distance between adjacent images significantly differs, the feature matching techniques outperform.

Estimating the egomotion of the vehicle via VO algorithms is superior to wheel-based odometry, because the latter becomes invalid for the legged robots. Besides this, VO delivers more accurate estimation of vehicle trajectory compared to the wheel-based odometry system (Scaramuzza and Fraundorfer 2011).

2.2.4 Loop-closure Detection

While feature tracking deals with the local consistency of a map, the loop-closure detection enforces global constraints on the topological structure of the map. It is often referred to as a *global data association*. A loop-closure event can be described as a re-observation of the landmarks that were observed long ago in the past (Bailey and Durrant-Whyte 2006). Hence, the vehicle re-visiting a previously visited (familiar) location must be able to recognize and associate the features observed at a query location with the learned features (map). Upon detecting a loop-closure, the location information is fed to the mapping algorithm to correct the global position of the robot (Scaramuzza and Fraundorfer 2011), as shown in Figure 2.4.

The loop detection is conventionally tackled using the gating or multi-hypothesis approach (Bailey and Durrant-Whyte 2006). For this purpose, the correspondences are searched between the query image's features and the features in the map. The gating method sets an uncertainty bound on the predicted landmark's position. If the error between the observed and predicted position of the landmark is within an acceptable bound, the landmarks are marked associated. The shape of the validation

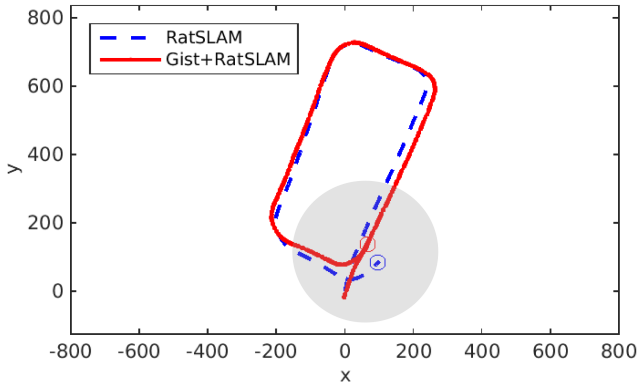


Figure 2.4: A loop-closure event: Gist+RatSLAM (Kazmi and Mertsching 2016a) algorithm has correctly detected the loop location on re-visiting the previously visited place. This information is leveraged for correcting the position errors and generating a consistent trajectory of the traversed path. By contrast, RatSLAM (Milford and Wyeth 2008) did not recognize this location due to weak data association – note the deviation from the loop location.

gate, i.e., the region around the predicted landmark position, is defined by a distance measure. A common choice is a Mahalanobis distance:

$$\text{dist}\left(\mathbf{z}_t^{(i)}, \widetilde{\mathbf{z}}_t^{(i)}\right) = \left(\mathbf{z}_t^{(i)} - \widetilde{\mathbf{z}}_t^{(i)}\right)^\top \mathbf{S}^{-1} \left(\mathbf{z}_t^{(i)} - \widetilde{\mathbf{z}}_t^{(i)}\right)$$

where $\mathbf{z}_t^{(i)}$ and $\widetilde{\mathbf{z}}_t^{(i)}$ are the observed and predicted positions of the i^{th} landmark, respectively, and $\mathbf{S}$ is a covariance matrix in the landmark's location. The gate is centered at the prediction $\widetilde{\mathbf{z}}_t^{(i)}$. On the other hand, multi-hypothesis driven data association maintains a disjoint set of loop and no-loop hypotheses over the perceptually similar locations. Each hypothesis is tracked separately forming a tree-like branching. Nonetheless, the above-mentioned methods suffer heavily in case of a poor repeatability – the condition when a landmark is not detected at a later time due to bad lighting or occlusion (i.e. hindrance to a landmark's visibility). Moreover, with the increasing scale of the map, these methods become computationally demanding, as one has to keep all the landmarks in map or track multiple hypotheses over the longer excursions.

To deal with the aforementioned shortcomings and the computational issues, the appearance-based methods have become popular in robotics community, see Section 2.5.2, which is the principal focus of this research.

2.3 Representing the Observed World

To operate autonomously in an unknown environment, a robot is supposed to build the representation, known as a *map*, of the observed world. A map can be

more precisely described as *"... the representation of a set of connected places which are systematically related to each other by a group of spatial transformation rules."* (O'Keefe and Nadel 1978, pp. 86).

A map encodes the routing information to reach a goal or plan paths on the basis of connectivity among places. Whilst the words "location" and "place" have similar lexical meaning, their connotative definition varies. For example, a location is a precise geometric entity in a Cartesian space which is referenced by coordinates, whereby a place is a humanly description of a location based on its characteristics, such as mountains, landscapes, etc. (Boehm and Petersen 1994).

The information contained in a map essentially depends on the type of the sensor deployed for map generation. Some important sensor-specific aspects for map building are summarized in Table 2.1. Indeed, each sensor perceives the environment differently and so provides exclusively distinct details of the surroundings. Moreover, every sensor has certain operational limitations, which must be taken into consideration to avoid inconsistencies in the map. For instance, the range sensors, i.e., laser range finder and sonar, deliver erroneous distance measurements in presence of atmospheric heat.

Table 2.1: Sensor-specific Factors that Influence the Map Estimation

Aspect	Range Sensors	Cameras
Features	Line, corner, etc.	Keypoint, patch, object, etc.
Noise source	Temperature	Illumination changes
Field of View	Wide	Narrow
Perceptual Aliasing[†]	High	Relatively low
Type of the Map	Occupancy grid	Sparse metric or topological
Map Management	CPU / Memory intensive	CPU / Memory efficient

[†] False detection of an unknown location as a familiar location due to insufficient or indistinguishable features. Range sensors are prone to such problems as primitive shapes, e.g., lines and corners, are frequently found in the structured environments.

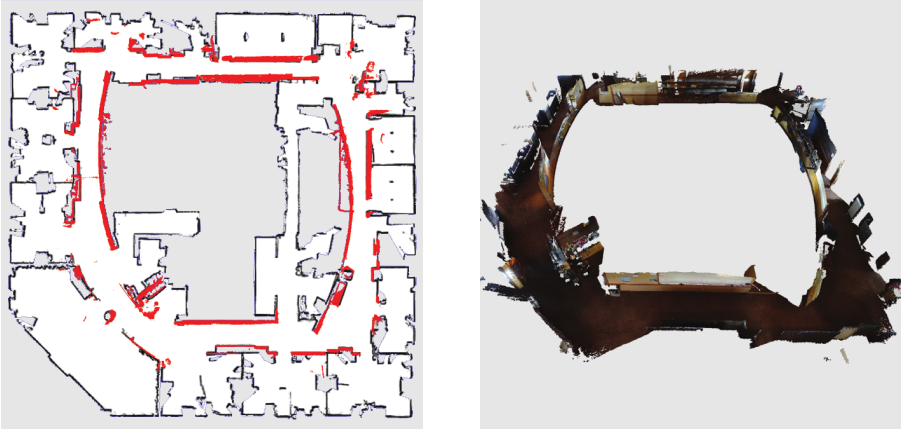
In general, maps are categorized as metric or topological.

2.3.1 Metric Maps

Metric maps represent the environment in a two- or a three-dimensional Cartesian space. Two common types of metric representations are: *Occupancy grid* and *landmarks-based* maps (Siciliano and Khatib 2008). In the former case, a map is obtained from range sensors. The occupancy map divides the space around the robot into cells, labeling each cell as free or occupied. Hence, the constructed map possesses the geometric layout of the free space and obstacles along the traveled path. This

makes the occupancy grid maps very useful for planning path and avoiding static obstacles. However, building such a representation of a large-scale environment is computationally demanding, especially the cost of updating the map to filter out dynamic objects.

The landmark-based map is typically built using imagery. To this end, the map is represented by a sparse set of point-like features that are tracked in the successive image frames. The camera motion and 3D position of the selected feature points are recovered from the images captured between the robot movements. Recently, the advent of RGB-D sensors has augmented the camera technology with the pixel-wise distance measurements to produce dense metric maps, though their precision is limited compared to laser range finders. Figure 2.5 depicts the maps of Intel research lab environment that are created with laser and RGB-D data.



(a) Occupancy grid map **(b)** RGB-D map

Figure 2.5: The figure illustrates the different maps of the Intel research lab. The occupancy map is obtained using laser range scanner, while the RGB-D map is built using the kinect (Modified on (Henry et al. 2012)). The red area in (a) is where the RGB-D map accurately overlays the occupancy map.

2.3.2 Topological Maps

For many roboticists, the topological mapping is a de-facto standard in SLAM (Choset and Nagatani 2001; Milford and Wyeth 2008; Ranganathan et al. 2006). The pioneer work on topological mapping dates back to late 1980s (Kuipers and Byun 1988), which describes the environment in form of a graph. The nodes in the graph represent the distinctive nearby places and the arcs correspond to the travel edges between the places. In this way, topological maps characterize the physical space as an ordered organization of places, without the necessity of using the geometric data between places. An example of a topological map is a subway map of a city. A place is hence associated with other places based on the order in which they are traversed and/or the visual similarity of the nearby places. The topological maps are also

closely related to how biophysical systems maintain maps for navigation, because the interpretation of maps in a mammalian's brain (including humans) is non-metric (Wyeth and Milford 2009).

2.4 Classic Map Estimation Algorithms

From a classical standpoint, robotic mapping is studied as a least squares estimation problem, in which case the unknown variables are the sequence of robot poses and the map of the observed world. Recovering these unknown variables is a non-trivial task in the presence of motion errors and noisy sensor data. Conventionally this challenge has been addressed in two general forms: (1) statistical filtering and (2) graph optimization. With their foundations in state estimation theory, the filtering methods track the optimal robot pose and the map in an incremental manner. To this end, a joint posterior is estimated over the most recent pose of the robot and the map of the landmarks. By contrast, the graph-based approaches model the SLAM problem as a sparse graph of poses. Whereas a joint posterior probability is estimated over the entire trajectory of the robot poses using the non-linear constraints optimization.

2.4.1 Statistical Filtering

In Section 2.1, we introduced a general definition of the online SLAM problem, which is described as a joint posterior distribution of the most recent robot pose and the map. Here, one can essentially make a choice on the distribution modeling. This segregates the filtering methods into two categories: (1) Kalman filtering, and (2) Particle filtering. Amongst the two frameworks, the former represents the posterior distribution with a parametric model, whereas the latter adopts a nonparametric approach to approximate the distribution.

Kalman Filtering

Kalman filters represent the joint posterior of the robot pose and the map of land–marks with a discrete-time multivariate Gaussian distribution:

$$p_{\mathcal{X}|\mathcal{Z}}(\mathbf{x}_t, \mathcal{M} \mid \mathcal{Z}_{1:t}, \mathcal{U}_t) = \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$$

The mean vector $\boldsymbol{\mu}_t$ contains the best pose of the robot up to present time and the position of landmarks in the map. The covariance matrix $\boldsymbol{\Sigma}_t$ accounts for the uncertainty or error in the current estimate, provided that the motion errors and the measurement noise are uncorrelated in time – a white Gaussian noise assumption. In case of a point-landmark, the dimensionality of the mean vector would be $3+2M$,

as we need three variables to represent the pose of the robot and two variables to represent the position of the M landmarks in the map. The covariance matrix is positive semi-definite and has a size $(3 + 2M) \times (3 + 2M)$.

The map estimation process relies on the motion and measurement models, see Eq. (2.2) and Eq. (2.4), respectively. These models are expressed as vector-valued non-linear functions, which makes the Kalman filtering infeasible to estimate the posterior distribution. Instead, it is common to use the variants of the Kalman filter, namely Extended Kalman Filter (EKF) or Unscented Kalman Filter (UKF). The EKF retains the first-order Taylor series approximation of a non-linear function, which is only effective for small camera rotations. For more significant changes in a vehicle's heading, the assumption of the local linearity violates. Hence, it is worth considering the higher-order terms for approximating the non-linear functions. In that vein, the UKF gives the third-order approximation of the posterior mean and the covariance based on a sampling approach.

Kalman filter-based methods (Dissanayake et al. 2001; Smith and Cheeseman 1986; Smith et al. 1990) recursively estimate an optimal pose of the robot and positions of landmarks with the aid of sensory observations. The algorithm first predicts the current robot pose and the landmarks' positions via known control inputs. Thereon the measurements are fused to minimize the prediction errors. In this process, the landmarks retaining high weights, aka Kalman Gain, contribute more towards updating the state of the robot and the map. As a result, the mean vector describes an optimal state estimate up to present time, while the covariance matrix explains the uncertainty in estimation.

If an optical camera is the only source sensor for robotic mapping, estimating the depth of the feature points is paramount. For this purpose, the local image features are observed over the consecutive robot movements and their 3D positions are recovered using motion parallax or bundle-adjustment algorithms (Yousif et al. 2015). In Kalman filtering, it is vital to initialize the state vector with the positions of the newly observed feature points, including their error in the covariance matrix. A workaround to deal with this concern is to estimate the position of the feature points based on the known size of a single object in an environment, such as (Kazmi et al. 2012), and initialize the landmarks' positions with low uncertainty values (Davison et al. 2007). As knowing the size of objects in advance is not always feasible, a group of researchers proposed alternative landmark initialization strategies (Bailey 2003; Civera et al. 2008; Clemente et al. 2007). The details of these and other related techniques are summarized in (Mirabdollah et al. 2016).

Over the longer excursions, Kalman filters do not deliver real-time performance for mapping tasks. This shortcoming originates from the fact that in large-scale

environments more landmarks must be included in the state vector, which results in an ever increasing size of the covariance matrix. Thereby, manipulation of the big matrices (multiplication and inversion) is the most expensive part of Kalman filtering – quadratic in the number of landmarks in the map.

Particle Filtering

Approximating the state of a dynamical system with a unimodal Gaussian distribution is a stark assumption, if the true state of the system is non-Gaussian. One solution to this challenge lies in the Monte Carlo simulation method, which estimates the posterior distribution of the system from a set of weighted samples, called particles. Particle filtering is a sequential Monte Carlo technique for approximating an unknown distribution of a system. The key advantage of particle filtering is its proficiency to deal with non-Gaussian or multi-modal distributions, including non-linearities in the motion as well as the measurement models. For a large number of particles, the approximation becomes a more accurate estimate of the unknown distribution (Siciliano and Khatib 2008).

It is not trivial to employ particle filtering in a SLAM context, as particles grow exponentially based on the number of landmarks in the map and the length of the path traversed by the robot. Hence, dealing with the high-dimensional feature vectors (e.g. 100 dimensions) is not amenable. As a result, particle filtering is frequently considered only for trajectory estimation. For map building, low-dimensional Kalman filters are applied for each landmark in the map, which enhances the timing efficiency and reduces the memory usage. This trick to use a particle filter for SLAM is known as Rao-blackwellization (Doucet et al. 2000), which is formalized as FastSLAM (Montemerlo et al. 2002). At an arbitrary time instant k, the algorithm maintains N_p particles, whereas each particle comprises the robot path $\mathcal{X}_k^{(n)}$ and M bivariate Gaussians $\left(\boldsymbol{\mu}_{k,m}^{(n)}, \boldsymbol{\Sigma}_{k,m}^{(n)}\right)$, one for each landmark in the map, as shown below:

$\mathcal{X}_k^{(n)}$	$\boldsymbol{\mu}_{k,1}^{(n)}$	$\dots$	$\boldsymbol{\mu}_{k,M}^{(n)}$	$\boldsymbol{\Sigma}_{k,1}^{(n)}$	$\dots$	$\boldsymbol{\Sigma}_{k,M}^{(n)}$

where $m \in M$ and $n \in N_p$ are indexes of the landmark and particle numbers, respectively. The algorithm starts with a pre-defined number of uniformly weighted particles, each representing a hypothesis of the robot pose and the landmarks' positions, as expressed above. For a given motion command, a new robot location is generated for each particle using the motion model and the map of the environment is predicted. Upon receiving the sensor observations $\mathcal{Z}_k$, the errors in the predictions are corrected. At this stage, the *importance weight* $w_k^{(n)}$ is calculated for every particle in the light of new observations:

$$w_k^{(n)} = \mathcal{N}\left(\mathcal{Z}_k \middle| \mathbf{x}_k, \boldsymbol{\mu}_{k,m}^{(n)}, \boldsymbol{\Sigma}_{k,m}^{(n)}\right)$$

The weights assigned to the particles are normalized so that $\sum_{n=1}^{N_p} w_k^{(n)} = 1$. Next, new particles are drawn with replacement from the existing particles, called as *resampling*. In this procedure, the probability that a particle gets selected depends on its importance weight. The key idea behind this process is to enforce the survival of the fittest mechanism, i.e., the particles with small position errors survive longer. Whereas, low weighted particles gradually vanish away due to large deviations from the observations. As a result, particle filter keeps multiple maps and the fittest particle is said to represent an optimal map of the environment (Bekris et al. 2006; Fox et al. 1999). The consistency of the built map strongly relies on the number of particles used to estimate the robot's trajectory. Moreover, multiple traversals to the previously visited locations cause particle depletion, which in turn leads to degenerated maps. A technical walk-through of particle filtering in a robotic mapping domain, the challenges and their possible solutions are discussed in (Montemerlo 2007; Siciliano and Khatib 2008; Van Der Merwe et al. 2001).

2.4.2 Pose-graph Optimization

In contrast to filter-based methods, the graph-based techniques (Grisetti et al. 2010a; Gutmann and Konolige 1999; Lu and Milios 1997) estimate the joint posterior over the trajectory of the robot poses $\mathcal{X}_t$ and the map $\mathcal{M}$:

$$p_{\mathcal{X}|\mathcal{Z}}(\mathcal{X}_t, \mathcal{M} \mid \mathcal{Z}_{1:t}, \mathcal{U}_t) \tag{2.7}$$

To this end, robot poses and landmarks are represented as nodes in a graph. The links between the nodes, called as edges, describe the spatial constraints between the poses and the landmarks. This concept is illustrated with a toy example in Figure 2.6. An edge between two arbitrary poses $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ encodes the robot's movement given by odometry $\mathbf{u}_{k+1}$. Another type of edge information comes from the observation of a landmark $\mathbf{z}_k^{(i)}$ with respect to the pose configuration $\mathbf{x}_k$. Hence, the edges correspond to the non-linear constraints on the structure of the graph. The relaxation of these constraints on the graph yields the best estimate of the robot's trajectory and the map of the environment.

In general, a graph-based SLAM algorithm comprises two steps: (1) graph construction and (2) constraints optimization. The graph construction part is the front-end and relies primarily on the type of sensory data. Note that the representation details about an environment are not a contemporary part of the graph. They are rather appended to the graph to express the environment as a sparsely metric or occupancy map. To illustrate the graph construction, let us re-visit the graph depicted in the toy example. This graph can be expressed in form of a matrix, as shown in the Figure 2.6. There exist two types of constraints in the graph, namely *odometry* and

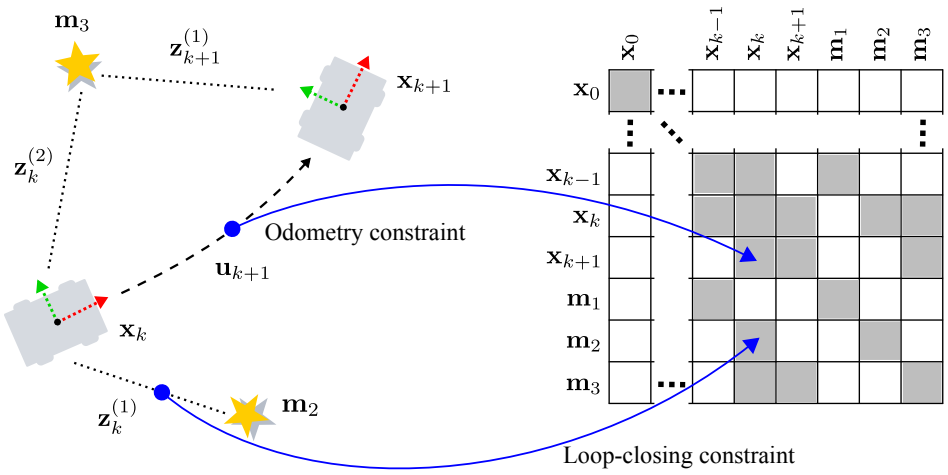


Figure 2.6: Matrix representation of the pose-graph: all the gray blocks in the matrix are constraints, of which the odometry constraints appear along the main diagonal, whereas the constraints due to landmarks' observation are the off-diagonal elements. For instance, the landmark $\mathbf{m}_2$ is related to the pose configuration $\mathbf{x}_k$ and so the corresponding element is filled with the real value. Note that the constraints between $\mathbf{x}_{k-1}$ and $\mathbf{x}_k$ are set in accordance with Figure 2.1.

loop-closure constraints. In this example, the odometry constraints appear only along the main diagonal in the matrix. E.g., the motion of the robot from the pose $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$ is marked with a value in the matrix, depicted as a gray box. This value could be any real number, such as Euclidean distance recorded by the odometry. The loop-closing constrains are due to the landmarks, which are marked in the matrix relative to the poses.

Note that the pose-graph matrix is sparse and symmetric about the diagonal. One can envision the graph as a spring-mass model (Golfarelli et al. 1998) in which an edge corresponds to the spring and the measurement error is proportional to the elasticity of that edge. The arrival of new measurements propagate the correction through the connected springs and so the system adapts itself. Applying this analogy to the SLAM problem lead us to finding out the state of minimum energy, i.e., the pose configurations with the least error. It is equivalent to finding the mode of the posterior distribution Eq. (2.7), known as a *maximum a posteriori (MAP)*:

$$\mathcal{X}_t^*, \mathcal{M}^* = \underset{\mathcal{X}_t, \mathcal{M}}{\arg\max}\ p_{\mathcal{X}|\mathcal{Z}}(\mathcal{X}_t, \mathcal{M} \mid \mathcal{Z}_{1:t}, \mathcal{U}_t)$$

Analogous to the filtering methods, the graph-based SLAM assumes white Gaussian noise in observation of landmarks. Thus, the above expression resolves to a sparse system of equations of quadratic form (Siciliano and Khatib 2008). The solution of such a system can be determined using optimization algorithms, ranging from the direct matrix decomposition, such as QR or Cholesky decomposition, to incremental optimization, e.g., conjugate gradient method (Konolige 2004), and others. Certainly, the iterative local linearization of the posterior distribution and revisiting the history

of data during the map building process makes the graph-based procedures superior to the filter-based counterparts.

The first offline formulation of the graph-based SLAM was introduced in late 90s by Lu and Milios (Lu and Milios 1997). Later this algorithm was further developed by Konolige (Konolige 2004) to improve the computational performance of loop-–closure and map registration processes. However, this family of algorithms did not gain popularity until 2006, when Thrun and Montemerlo (Thrun and Montemerlo 2006) introduced a variable elimination method to reduce the complexity of the graph optimization procedure. To-date, there exist several graph optimization al–gorithms: $\sqrt{\text{SAM}}$ (Dellaert and Kaess 2006), iSAM (Kaess et al. 2008), stochastic gradient descent (Grisetti et al. 2009; Olson et al. 2006), Sparse Pose Adjustment (Konolige et al. 2010), HOG-Man (Grisetti et al. 2010b) and g^{2}o (Kümmerle et al. 2011). For an in-depth understanding of the pose-graph algorithms, we defer the reader to the tutorial (Grisetti et al. 2010a).

2.5 Place Recognition

The correct data association (Section 2.2.4) is by far the biggest challenge to a long-term navigation. It would have been a trivial task, if the pose estimation er–rors were de-correlated in time. Over the past few years, the filtering frameworks (Section 2.4.1) have been under continual attention of the researchers. These ap–proaches constantly track the robot pose and the landmarks' coordinates in the map, so that the error on re-visiting the previously traversed routes does not blow out of an acceptable bound. Nonetheless, associating features in a metric map space remains susceptible to wrong matches. Moreover, this nature of comparison gets computationally intense with the ever-increasing size of the explored environment.

With the recent advancements in computer vision, the place recognition has become one of the most cited topics in the domain of robotic mapping, particularly for loop detection (Artac et al. 2002; Cummins and Newman 2011; Krose and Bunschoten 1999; Milford and Wyeth 2012; Schubert et al. 2023; Tsintotas et al. 2022; Ulrich and Nourbakhsh 2000). Place recognition is a process of identifying a query image in the large database of learned images.

2.5.1 Defintion of a Place

A place may be assigned various definitions according to the context. For instance, at a city level, a university could be geotagged as a place, while within the university

different buildings (or departments) can be considered as multiple places, and the interpretation varies further at the room level.

One way to define a place could be geotagging the images at specific GPS coordinates, and refer to each GPS location as a place. But GPS data is not always available, such as in tunnels, dense vegetation areas or indoors. In this case, new places can be added based on certain spatial or temporal constraints, e.g., when a robot travels a specific distance or a certain amount of time has elapsed since the last recording of images, respectively. Alternatively, one can exploit the visual description of images to describe a new place.

2.5.2 Emergence of Appearance-based Mapping

Krose and Bunschoten (1999) presented the notion of modeling the observed world in the space of images (”visual appearance"). Later, Ulrich and Nourbakhsh (2000) utilized color histograms to capture the appearance of places and studied the role of place recognition to address the robotic mapping task, referred to as *appear–ance-based mapping*. As the name suggests, the appearance-based paradigms dis–criminate the places from one another based on visual properties contained in images. To this end, an image recorded at a certain time is characterized by a single vector, known as a *global descriptor*. There exist several ways to capture the visual description of a place, see Section 2.6. Given the global description of a query image, its perceptual similarity to other places is learned by means of a scoring function, without the need of any geometric information.

As a result, instead of associating a bulk of features in a Cartesian map, the visited and unvisited places are distinguished in the space of visual appearance, as shown in Figure 2.7. Here, the scoring function ϕ may be a matching metric, such as a Euclidean or Manhattan, etc., a machine learning algorithm or a belief function (Cummins and Newman 2011; Sivic and Zisserman 2003). Indeed, the localization of a vehicle in the visual space is analogous to the way biophysical systems, e.g., humans, learn and interpret diverse environments (Wyeth and Milford 2009), which is one of the primary questions we investigated in this research.

2.5.3 Data Association – in a Metric Map vs. Image Space

In scenarios where the metric representation of an environment is needed, the appearance-based methods can serve as a loop detection module. Table 2.2 sum–marizes the principal loop detection methods with their strengths and weaknesses. The comparison in the table originates from the results demonstrated in (Williams

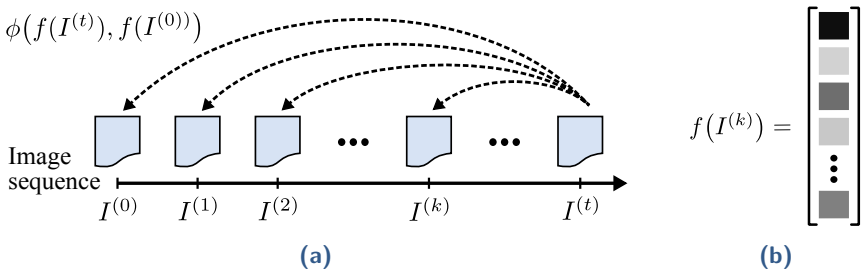


Figure 2.7: At a present time t, the visual description of the image frame $I^{(t)}$ is obtained using some feature extractor function $f(.)$ and its similarity score ϕ is calculated with the descriptors of all the previous frames, except the recent ones. The shades of gray in the global descriptor refer to different real values.

et al. 2009). Searching the correspondence points in the map, as explained in Section 2.2.4, is referred to as a *map-to-map* loop detection (Clemente et al. 2007). Whereas, an *image-to-map* correspondence (Williams et al. 2008) is done between the local features extracted from a query image and the features in a coordinate space of the map. An *image-to-image* correspondence (Cummins and Newman 2011) is indeed an appearance-based mapping, as the data association is defined in the feature space.

Note that image-to-image and image-to-map techniques are fast compared to pure map-based loop detection. Nevertheless, if the correspondences are sought in the map, as is the case with image-to-map and map-to-map methods, the scalability is limited to small environments due to high memory needs and enormous comparisons among the feature points.

Table 2.2: Qualitative Comparison of the Popular Loop Closing Methods

Comparison Factors	Image-to-Image	Image-to-Map	Map-to-Map
Descriptor	Global	Local	Local
Suitable Map	Topological	Sparsely metric	Dense metric
Memory Requirements	Low	High	High
Loop Detection Efficiency	Fast	Fast	Slow
False Detections	Few	None	Many
Training	Offline	Online	Not Needed
Environment Scalability	Large-scale	Small-scale	Small-scale
Failure Mode	Bad threshold	Inaccuracies in map	Few Features

This highlights the superiority of the image-to-image correspondence in large-scale environments, though the requirement of prior training and selection of a suitable threshold (parameter tuning overhead) restrain the autonomy of vehicles in unknown environments. These and many other shortcomings of the image-to-image approaches are addressed in the scope of this research, the details of which are explored in the latter chapters.

2.5.4 Evaluation Criteria

In the space of visual appearance, a metric measurement of the location error is void. Hence, for assessing various characteristics of a place recognition system, it is necessary to describe the evaluation criteria. This will serve as a basis to test the accuracy of appearance-based methods and compare their performances. Before we formally state the performance measures, let us consider the following example to understand the types of errors that a recognition system may commit:

> *One pleasant day, John sets a trip to mountains. While passing by a wild forest on his way, he stops the car on the road shoulder to test the coolant fluid in the engine. After stepping out, he sees an animal few hundred meters away. He ignores it, assuming as if it is a wild cat, but a glance at the signboard warns him of a danger of encountering panthers.*

Given that John has read the warning board, what are the odds that the observed animal is a cat? Can the consequences be serious, if John's observation is wrong? To answer the questions, we make a hypothesis that the animal is not a cat. If the seen animal is not a cat but John has observed it as a cat (*type I* error), then John's hallucination of a cat has put his life at stake. Such a mistake is termed as *false positive (FP)*. However, if the animal is a cat in reality but John gets an impression of a panther (*type II* error), then he will take the safety measures, such as going back into the car. Although he will lose time due to the false belief, the loss is not as serious as type I error. This kind of error is named as *false negative (FN)*. All the correct cases are the examples, when he correctly recognizes the animal as a cat (*true positive: TP*) or a panther (*true negative: TN*). To make the concept more explicit, all the case are depicted as a confusion chart in Figure 2.8 (left).

Precision–Recall Score

Using the above information, one can quantify how vigilant John is in general, provided his behavior and responses are observed on multiple times (or similar occasions). A well-known measure to evaluate the performance of a recognition system is a *precision–recall* score[2], which has its origin in information retrieval (Rijsbergen 1979). In general, precision is a measure of accuracy of a system to retrieve the relevant results without picking up the garbage as relevant, whereas recall is the degree to which a system retrieves the relevant results and does not miss any.

[2]The precision–recall scores exist in the range $[0, 1]$, but the scores are usually multiplied by hundred to interpret the result in percentage (%) – as is the case here.

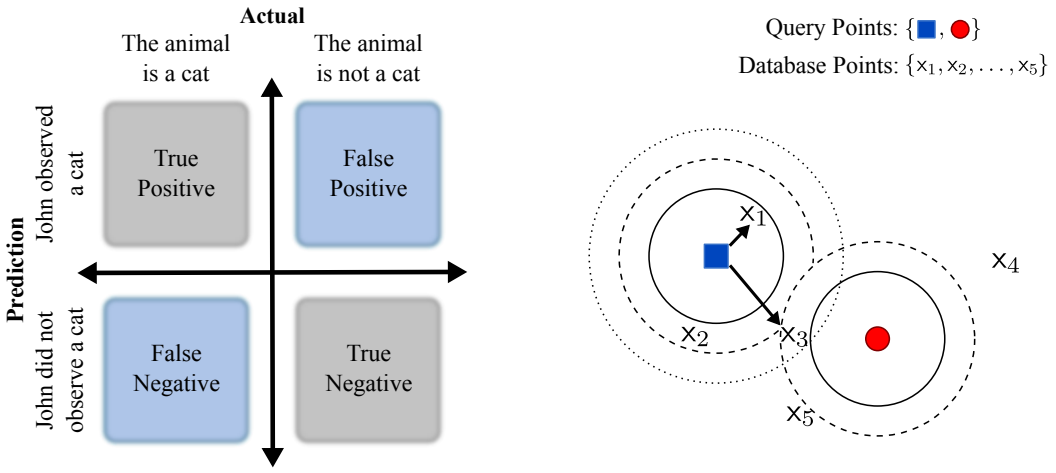


Figure 2.8: (Left) Confusion matrix for binary classification: actual cases along the columns are the real world situations, while the predictions make-up the rows. (Right) Query search in a database of points at different thresholds, depicted as circular regions around the query points.

With regard to our recent example, precision (P) is a fraction of cases in which John correctly detects the cat (true positive) out of all the cases in which he recognizes the observed animal as a cat (true and false positives):

$$\text{P (\%)} = \frac{\text{True positive cases}}{\text{True positive cases} + \text{False positive cases}} \times 100 \tag{2.8}$$

Recall (R) is the fraction of all the correct detections of a cat (true positive) among all the cases in which John correctly recognizes the cat and not confuses it with a panther (true positive and false negative).

$$\text{R (\%)} = \frac{\text{True positive cases}}{\text{True positive cases} + \text{False negative cases}} \times 100 \tag{2.9}$$

From a technical perspective, here place recognition, we are interested to find out the best representative(s) of a query image in a learned database. Consider the two query points, a blue rectangle and a red circle, as depicted in Figure 2.8 (right). To keep the things simple, assume that the points lie on a two-dimensional plane and the matching strategy is a Euclidean distance between the query and database points. Also assume that the correct best matches of a query point are its two nearest neighbors (*ground truth*).

Given the blue-rectangle query, we calculate its distance from all the points $\mathbf{x}_i$ in the database, and select $\mathbf{x}_1$ based on specified threshold criterion (solid line), which is a TP prediction. At the same threshold setting, the matching scheme does not retrieve $\mathbf{x}_2$ for the blue-rectangle and finds no match for the red-circle query. Missing the correct matches are the mistakes of the algorithm and they are regarded

as FN predictions. By pushing the threshold limit to a larger distance (dashed line), one obtains two TP candidates for the rectangle query point and one TP match for the circle, but the missing $\mathbf{x}_5$ is a FN prediction. Note that bigger radii help collecting more matches, but the odds of FP predictions increase because the decision boundaries around query points jumble up. For instance, if the distance criterion is further relaxed to a dotted line, as it is shown for the blue-rectangle case, $\mathbf{x}_3$ can be associated with both the rectangle and circle query points. In the present setup, assigning $\mathbf{x}_3$ to the rectangle is a FP prediction, which according to the ground truth, does not belong to the rectangle query. Note that $\mathbf{x}_4$ remained a true negative (TN) for all of the above thresholds.

Figure 2.9 illustrates the variation in precision and recall values at different threshold settings: T_1 (solid line), T_2 (dashed line) and T_3 (dotted line). Each threshold setting corresponds to a precision–recall coordinate pair on the curve. The obtained curve can be used to interpret the robustness of a recognition system. Ideally, a system is said to have the best performance, when both the precision and recall scores are 100%. However, the ideal conditions are hard to fulfill in real-life scenarios due to sensor noise, perceptual similarity among scenes and other environmental conditions, e.g., bad weather or seasonal changes.

In the present example, the 25% recall score at 100% precision is too low. There could be a debate on which of the other two cases, i.e., $(75\%, 100\%)$ and $(100\%, 80\%)$, are the best. Infact, it depends on the demanding nature of an application. For instance, in mission critical systems, such as autonomous navigation, banking, etc., low precision values are unacceptable. In principal, low precision is the consequence of high false positive predictions. Such mistakes can lead to a catastrophic failure of a navigation system or financial loss. By contrast, searching a query image on a

	T_1	T_2	T_3
FPs	0	0	1
TPs	1	3	4
FNs	3	1	0
P (%)	100	100	80
R (%)	25	75	100

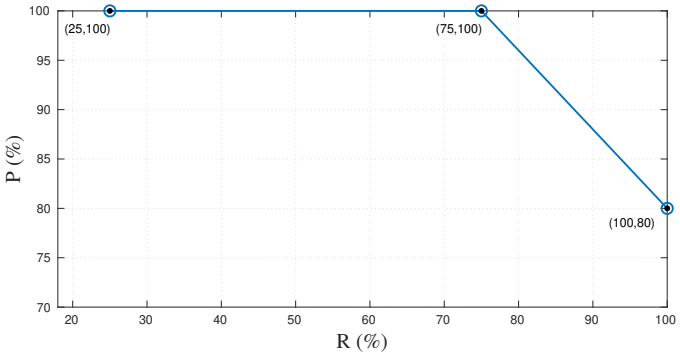


Figure 2.9: (Left) The scores listed in the table relate to the query point example of Figure 2.8, at different threshold settings (T_i). (Right) Observe the scale along $x-$ and $y-$axes. The precision–recall curve corresponds to the coordinates listed in the table.

favorite search engine and still getting some false candidates is relatively acceptable, as the user may further choose the most appropriate images.

F-measure

F-measure (or F-score) is often named as F_β-measure. It was invented to put the importance weight on the precision scores relative to the recall values (Chinchor 1992). Hence, F_β-measure can be regarded as a weighted average of the precision–recall scores. It is calculated as follows:

$$F_\beta = (\beta^2 + 1) \cdot \frac{\mathrm{P} \cdot \mathrm{R}}{\beta^2 \cdot \mathrm{P} + \mathrm{R}} \tag{2.10}$$

Typically, the value of β is set empirically. Each coordinate pair on a precision–recall curve has a corresponding F_β-score. Hence, instead of maximizing the two quantities at a time, one has to find a single highest score. For $\beta = 1$, the precision and recall are equally weighted. Hence, F_1-score is referred to as a harmonic mean of the precision and recall scores. One drawback of using F_1-score is that it does not distinguish the cases where precision and recall values interchange. For instance, the F_1-score for P: 100% and R: 75% remains the same if precision drops to 75% and recall becomes 100%. When $\beta < 1$, F-measure assigns more importance to precision than recall, while $\beta > 1$ puts emphasis on recall scores compared to precision.

2.6 Visual Description of a Place

A naive visual description of a place could be the set of all the features detected in an image. So, the similarity between any two images can be regarded as the number of matched local descriptors. The computation cost of this comparison scales quadratically with the features count (Aly et al. 2009). Indeed, one can loosely bound the computations to a constant time, if the feature matching is done for a handful of images. However, it is quite untrue for the loop detection task, as one has to match the local descriptors for a massive collection of images in the database. Such an intense correspondence search degenerates not only the efficiency but also the accuracy of the system.

Recently, the global description of an image (aka whole-image description) has gained substantial importance for the recognition and image retrieval tasks (Lategahn et al. 2013; Li et al. 2008; Oliva and Torralba 2006; Sivic and Zisserman 2003; Torii et al. 2015). Unlike local descriptors, which are computed around a small neighborhood of the individual pixels (keypoints), a global descriptor is computed for an entire image and thus it encodes the visual properties, such as color, edge

distribution, etc., in an image as a single vector. The numerical information stored in the vector is regarded as a visual description of the place. Hence, a global descriptor summarizes the position invariant statistics of an image. A naive global description could be an intensity or a color histogram (Ulrich and Nourbakhsh 2000).

Describing images globally is robust to dynamic lighting conditions compared to a local description (Han et al. 2017; Kazmi and Mertsching 2019a; Lowry et al. 2016b). Although local descriptors are superior to the global descriptors for large viewpoint changes, their performance significantly suffers due to the moving objects, especially when the previously detected keypoints get occluded in an image. Furthermore, to detect the stable local features, one usually iterates over a set of correspondence points and selects those that minimize the re-projection error (Knopp et al. 2010; Nistér 2004; Turcot and Lowe 2009). This procedure is known as a *geometric consistency check*. A trade-off to enjoy the benefits of the local and global descriptions is to extract the global description from the large image segments and stack them up into a single vector, as also mentioned by Lowry et al. (2016b) and demonstrated in works (McManus et al. 2015; Sünderhauf et al. 2015b). The gist descriptor (Oliva and Torralba 2001) is also computed in quite similar way, see Section 2.6.3. Following sections discuss the frequently adopted global descriptors for appearance-based mapping.

2.6.1 Intensity-based Descriptor

The intensity-based description of a scene is sampled directly from the grayscale version of images. To this end, an input image is down-sampled to a very low resolution image followed by a patch normalization step (Milford and Wyeth 2012). The purpose of patch normalization is to adjust the abrupt intensity variations in a small window and thus making the final representation robust to the illumination changes. Finally, the resulting image is unrolled into a single column vector to represent the image's intensity as a global descriptor.

2.6.2 Vocabulary-based Descriptor

The vocabulary-based description of an image has its roots in the domain of information retrieval (Baeza-Yates and Ribeiro-Neto 1999). For computer vision applications, the vocabulary-based image representation receives popularity after (Sivic and Zisserman 2003) extended the concept of words (for document retrieval) to the visual words for image retrieval. Hence, a vocabulary-based image description is also referred to as a bag-of-words (BoW) or a bag-of-visual-words (BoVW) model.

A general procedure for training a visual vocabulary and representing an arbitrary image using the learned vocabulary is depicted in Figure 2.10. For vocabulary learning, first the features, such as SURF (Bay et al. 2008), ORB (Rublee et al. 2011), gist (Oliva and Torralba 2001), etc., are extracted from an exclusive set of training images that do not overlap. Following the feature extraction, the clustering algorithm, e.g., k-means or k-medoids, is applied, which quantizes the feature space into a group of points, called clusters, where each cluster centroid (or medoid[3]) represents a *visual word* in the vocabulary. Eventually, the number of cluster centroids (or medoids) determine the size of the trained visual vocabulary. Once the vocabulary is learned on the training data, any arbitrary image (query or database) can be described as a histogram of occurrences of visual words present in the image. Given an input image and the learned vocabulary, obtaining a vocabulary-based description of the image involves the following steps: (1) extract features from the image, (2) for each extracted feature, search the nearest visual word in the vocabulary using an appropriate distance measure, e.g., Euclidean or Manhattan distance, and (3) if the nearest word lies within the pre-defined threshold range, increment the counter of the corresponding bin in the histogram.

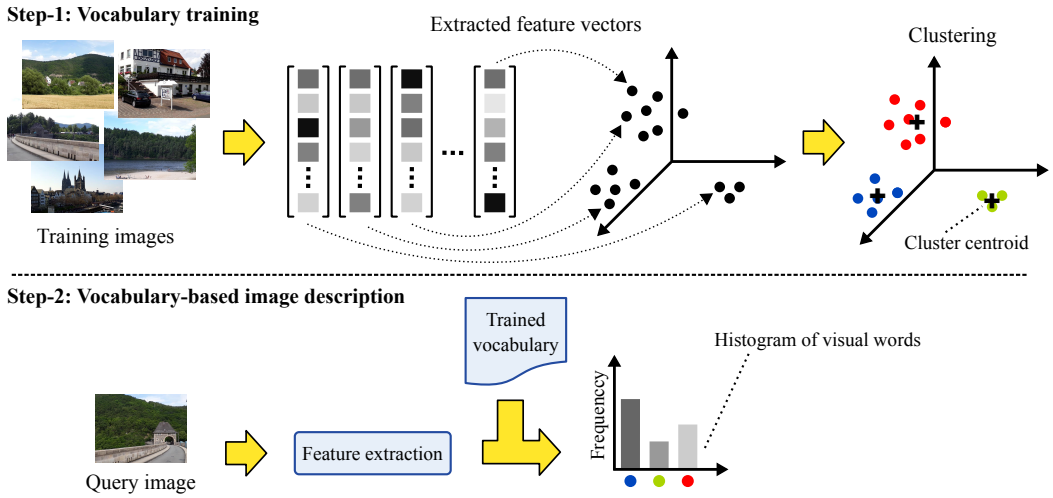


Figure 2.10: Vocabulary-based image description: First, the features are extracted from the training data and the clustering is performed. The cluster centers correspond to the visual words in the vocabulary. Later, for any arbitrary image, the features are extracted and the trained vocabulary is leveraged to represent the whole image as a histogram of visual words observed in it. If no feature is found for a particular visual word, the corresponding histogram bin will appear empty.

2.6.3 Gist Descriptor

Computing the gist descriptor of an image takes motivation from how humans obtain the visual description of the surroundings. The term "gist" refers to the amount

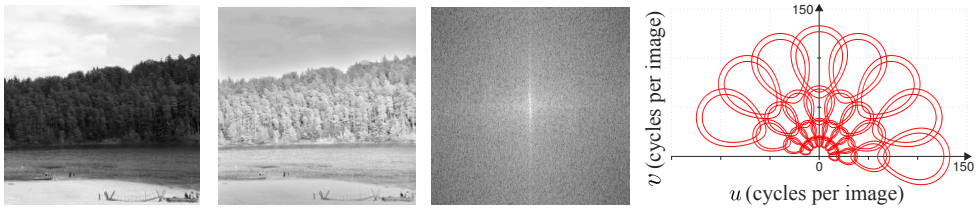
[3]Medoid differs from centroid (mean) in the sense that the cluster is represented by the point whose average dissimilarity to all the other data points in the cluster in minimum.

of information that a human vision system perceives within a small fraction of a second (Oliva and Torralba 2006). In the following discussion, we briefly discuss the procedure to extract the gist of a scene from an image, while the biological aspect of the discussion is presented in a later Section 2.7.1. From a computational standpoint, the gist of the scene can be extracted from an image by applying a bank of filters to the image and collecting the frequency[4] content at multiple scales and orientations. The frequency information extracted in this process describes the global layout of the scene in terms of perceptual attributes, such as a degree of natural content, openness, ruggedness, and others (Oliva and Torralba 2001).

To compute the gist features, a grayscale input image $I(x, y)$ is down-sampled, contrast normalized and cropped to a resolution of $N \times N$ pixels ($N = 256$), see Figure 2.11a and Figure 2.11b. The preprocessed image is transformed to the Fourier domain (frequency). Finally, the power spectrum of the image $|I(u, v)|^2$ (Figure 2.11c) is sampled using a set of transfer functions $H_i(u, v)$:

$$G_i(u, v) = |I(u, v)|^2 \; H_i(u, v) \tag{2.11}$$

where the transfer functions correspond to the Gabor filters at four different frequency bands, with eight orientations at each level.



(a) (b) (c) (d)

Figure 2.11: Main steps to compute the gist of a scene: (a) Input image (b) Output after preprocessing (256×256 pixels) (c) Power spectrum of the preprocessed image. In the original image, major brightness changes exist horizontally, validate visually by scanning the image from top to bottom. These changes reflect back in the power spectrum as a white dominant line along the vertical axis. (d) A bank of $\lambda = 32$ filters to sample the frequency content in images, where u and v are the frequency variables in Fourier domain.

Filter responses similar to Gabor filter can be modeled with the modulated Gaussian function (Oliva and Torralba 2001):

$$H_i(u, v) = e^{-v^2/\sigma_v^2} \left(e^{-(u-u_0)^2/\sigma_u^2} + e^{(u+u_0)^2/\sigma_u^2} \right)$$

[4] In contrast to a one-dimensional discrete time signal, a digital image is a two-dimensional spatial signal, in which the brightness (or intensity) levels change in vertical and horizontal directions. A complete dark to light transition (or vice versa) is referred to as a cycle. Hence, the spatial frequency in an image is defined as the number of transition from one brightness level to another per image (or millimeters).

Figure 2.11d shows the tiling of the filters H_i in Fourier domain. For the specified spatial frequencies u and v, the parameter u_0 determines the filter's center, whereas σ_u and σ_v tune the spread of the Gaussian along horizontal and vertical directions, respectively. After capturing the filter responses $G_i(u, v)$, the results are transformed back to the spatial domain using inverse Fourier transform:

$$g_i(x, y) = \frac{1}{N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} G_i(u, v)\, e^{j2\pi(ux+vy)/N} \tag{2.12}$$

Finally, the responses $g_i(x, y)$ are averaged individually over 4×4 image blocks, which results in a $D = 16 \times \lambda = 512$-dimensional vector of the gist features.

2.6.4 HOG Features

The HOG (histogram of oriented gradients) features (Dalal and Triggs 2005) have been mainly used to detect objects (in particular humans) in images. However, they can also be applied to an entire image to obtain the appearance of a scene in terms of distribution of edges (i.e. intensity gradients). To this end, image is convolved with the centered derivative mask. The resulting derivative image is then utilized to compute the gradient orientations and the magnitudes. To construct the global descriptor of an image, there are several design decisions to make, such as number of blocks (i.e. local detection windows) N_b in an image and the number of histogram bins N_h. These aspects have a considerable impact on the robustness of the HOG descriptor to illumination changes. For example, if the block size is too small, then it means we look for specific details in the detection window. In doing so, one makes the descriptor sensitive to certain image details. On the contrary, a large block size reflects general details in the image (in terms of distribution of edges). In the same vein, number of histogram bins make the descriptor sensitive to brightness changes. Hence, one should make a balance of block size and the number of histogram bins. In this research, we found that $N_b = 4 \times 4$ blocks and $N_h = 12$ offer a good trade-off between accuracy and efficiency (see Section 6.5.9).

To construct the global descriptor, we first pre-process the image, similar to the procedure described in Section 2.6.3. Thereafter, the image gradients are computed and histograms of the corresponding blocks are determined. This results in 16 histograms, describing the orientations of edges in the range $[0, 2\pi]$ radians. The value of a bin in a histogram corresponds to the sum of the magnitudes of gradients which fall within that bin. Concatenating all the histograms together results into a single $D = N_b \times N_h = 192$-dimensional global descriptor of the image. This descriptor can now be normalized using different methods, e.g., L1- and, L2-norms, whereas we choose the L1-norm, as the computation of a square root in L2-norm is

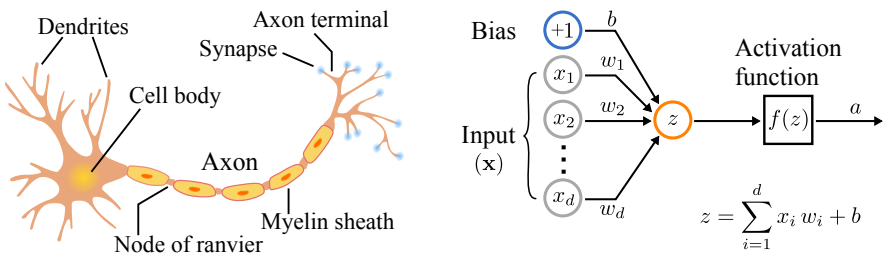
an expensive operation. Furthermore, in the experiments (Dalal and Triggs 2005, Figure 4(c)), authors do not observe significant performance drop for L1-norm.

2.6.5 CNN-based Descriptor

A convolutional neural network (CNN or ConvNet) is a type of artificial neural network, which emulates the biological processes in primary areas of cortical regions, see Section 2.7. Thus, a CNN represents the patterns in the input data hierarchically with the increasing order of complexity. An elementary logistic unit in a CNN is a neuron, which is an approximate model of a biological neuron. At a functional level, a neuron receives inputs at its dendrites in form of synapses from other neurons (or sensory organs). Where a synapse is an electrical impulse that a neuron exchanges with other neurons for communication. The input signals are integrated in the cell body and the response is propagated to other neurons through axon terminals (subject to activation of a neuron), see Figure 2.12a. By the same analogy, the design of an artificial neuron is characterized by three components: (a) inputs, (b) summation junction, called as processing unit, and (c) the activation function. A model of the artificial neuron is depicted in Figure 2.12b.

Let us consider that an d-dimensional input $\mathbf{x} = \{x_1, x_2, \ldots, x_d\}$ is fed to a neuron. Each input x_i is multiplied by a factor w_i and the result is aggregated as $z = \sum_{i=1}^{d} x_i\, w_i + b$. This result is then processed through an activation function $a = f(z)$ to get the final response of the neuron. Note that the bias term b is usually added to the total sum for adjusting the threshold value of the activation function.

There exist different choices for the activation function, such as sigmoid, ReLU (rectified linear unit), and others (Nair and Hinton 2010). When the multiple neural units are connected together, they represent a neural network. A neural network is



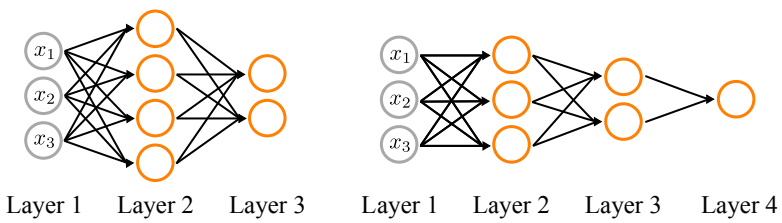
(a) Biological neuron **(b)** Artificial neuron

Figure 2.12: Artificial neuron is an approximate computational model of the biological neuron. The inputs in a biological neuron are the synapses received at dendrites from other neurons or sensory organs, which are combined in the cell body and the response is propagated to other neurons through axon. An axon is the outgoing fibre from the cell body. It is covered with the myelin membrane to protect the neuron's response and support fast neural transmission.

typically organized in multiple layers. All the neural units in one layer are connected to every unit in other layer, thus forming a fully-connected network of neurons. The first layer in the network is considered as input layer, as it feeds the input data to the next layers. The last layer in the network is referred to as an output layer and it generates the final response of the network. All the layers between input and output layers are referred to as hidden layers. Where the terminology "hidden" refers to the fact that these layers learn the unknown patterns from the input data. The interconnection of multiple layers in the network defines the network architecture. Figure 2.13 depicts an example three- and four-layer network architectures.

There exist variety of approaches for training a neural network, amongst them most well-known are feedforward and backpropagation methods. In feedforward training, the flow of information is only in one direction, i.e., from the input layer, through the subsequent hidden layers and finally reaching the output. The neurons in each layer of the network learn some complex functions or features as a linear combination of the output response of predecessor neurons and the corresponding weights on the incoming connections. On the contrary, the backpropagation method trains the network in forward and backward passes. The forward pass follows the procedure as it for the feedforward training, while in the backward pass, the network's output is compared to the desired output. The resulting error, given by some loss function, is fed back to adjust the weights at the input and hidden layers. This forward-backward process is repeated until the error between desired and the network's output gets sufficiently close to zero.

CNN share the similar properties as regular neural networks, but they assume that the input data is images. This assumption enables incorporating certain architectural changes in the network, such as each network layer is represented as a three-dimensional volume of neurons (width, height and depth), the neurons in one layer only connect to a small region in the predecessor layers instead of full connectivity between the layers. Such a design speeds-up the forward pass and reduces the number of network parameters to be trained, as the neurons now share the connections in a local pixel neighborhood. As a result, a CNN is a volume of activations from



(a) Three-layer network **(b)** Four-layer network

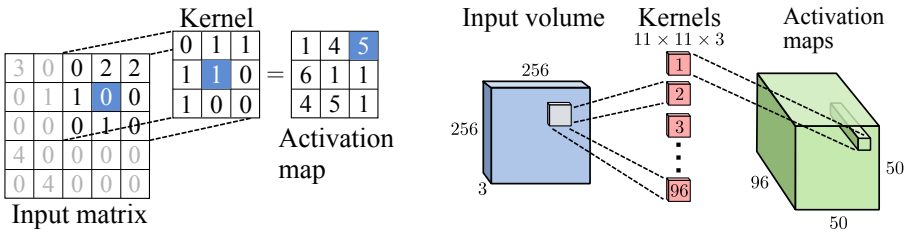
Figure 2.13: Example of different network architectures. As activation function is an inherent part of each neural unit, we did not draw it explicitly.

one layer to another. A simple CNN architecture comprises three types of layers: convolutional layer, pooling layer and fully-connected layer.

Convolutional Layer

In a convolution layer, an input matrix is convolved with a set of kernels, aka filters. The kernels are initialized with the random weights and they are learned from the training images, where the number of kernels (D) is selected by the user at network initialization time. Each kernel is applied across the width and height of the image, which results in an activation (or feature) map. An example convolution operation for a two-dimensional data matrix is shown in Figure 2.14a. The kernel size, here 3×3, describes the effective receptive field area around a pixel in the input matrix. Hence, a convolution operation is the sum of products of kernel coefficients with the corresponding matrix elements. The output of the convolution operation is another matrix, termed as an *activation map*, which contains the filter responses at each pixel position of the data matrix.

Stacking all the responses together forms a volume of feature maps, as shown in Figure 2.14b. The depth of the volume corresponds to the number of kernels used ($D = 96$). In a convolution operation, the width and height of the output are controlled by the parameters *zero padding* (P) and *stride* (S). Zero padding refers to the number of zeros or the null vectors (row/column) that are padded on the boundaries of the input before convolution. On the other hand, stride is the amount by which the kernel is slid horizontally and vertically on the input. Hence, a large stride value evaluates the kernel responses at fewer sample positions along the width and height of the image. In general, convolving an input matrix of size $N \times N$ with a kernel of size $K \times K$ yields another matrix of size $M \times M$, where $M = (N - K + 2P)/S + 1$. Note that P is typically related to K by $P = (K - 1)/2$. The figure illustrates convolution of a 256×256 pixels color image with 96 different kernels of size $11 \times 11 \times 3$, using a stride of five and without zero padding.



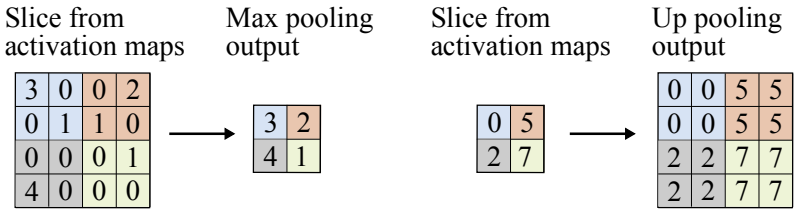
(a) Convolution operation **(b)** Volume of filter responses

Figure 2.14: An example of convolution operation for a two dimensional input: (a) illustrates the convolution of a 5×5 input with a 3×3 kernel. The same concept applies to convolving volumes (b), which in the present case is the convolution of 3D kernels with the color image. Each slice (50×50) along the depth of the output volume is the response of a particular kernel.

Pooling Layer

The pooling layer is usually inserted after the convolution layers. There are different types of pooling operations, but the most common among them are the *max pooling* and the *up pooling*. The purpose of the max pooling operation is to downsample the spatial size (width and height) of the representation learned by the convolutional layer (Scherer et al. 2010). This in turn reduces the number of network parameters and controls the overfitting. The pooling operation is defined separately on each depth slice. To this end, the filter regions are specified and the maximum value in that region is selected as an output. Figure 2.15a depicts the concept of max pooling for a filter of size 2×2 with a stride of 2. The size M of the output after max pooling is generally given by $M = (N - K)/S + 1$.

Besides max pooling (downsampling), the pooling layer is often used to upsample the input, referred to as up pooling operation. There exist various upsampling methods, namely nearest neighbor interpolation, bilinear interpolation, etc. An example of the up pooling operation for a 2×2 kernel using the nearest neighbor interpolation is demonstrated in Figure 2.15b.



(a) Max pooling **(b)** Up pooling

Figure 2.15: An example of pooling operations on a slice of convolution layer. Note that the pooling is applied independently to each slice of the activation maps of the predecessor layer.

Fully-connected Layer

The fully-connected layers are the last layers in a CNN. All neurons in this layer are connected to all the other activations in the preceding layer, which is same as depicted for regular networks in Figure 2.13. These layers generally posses abstract and task specific information, such as object recognition and image categorization. For instance, in case of an image categorization task, this layer carries the score assignments for each category.

Exploiting Network Features as a Global Descriptor

For extracting features from a CNN, the Places205-AlexNet network (Zhou et al. 2014) is selected in this work; the network architecture is depicted in Figure 2.16. The rational for using above-mentioned CNN is explained in Section 3.5. As we

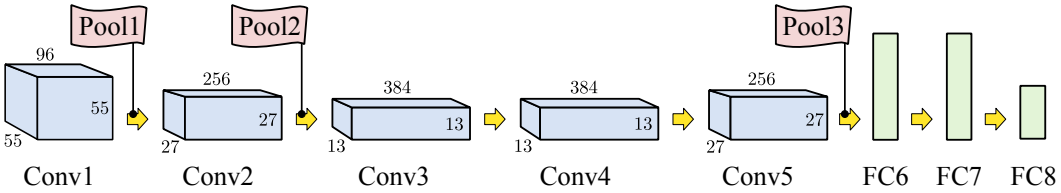


Figure 2.16: Architectural representation of the AlexNet (Krizhevsky et al. 2012). Note that max pooling operations are performed only at the output of Conv1, Conv2 and Conv5 layers. As a result, the spatial size of the subsequent layers is reduced. In original work, the volume at each convolution is separated into two chunks along the depth dimension (number of filters) to perform convolutions in parallel on two GPUs.

have exploited a pre-trained network for feature extraction, the technical details for training a network fall beyond the scope of this research. For a deeper insight, interested readers are deferred to the respective works (Jia et al. 2014; Krizhevsky et al. 2012). In the following discussion, we briefly outline the architecture of Places205-AlexNet and describe the procedure to leverage the network's features as a single global descriptor.

The Places205-AlexNet network share the same architecture as AlexNet (Zhou et al. 2014), except that the network has been trained with a focus on place recognition with a scene-centric database. There are eight learned layers in AlexNet, amongst them are five convolutional layers (Conv1, Conv2, ..., Conv5) and three fully-–connected layers (FC6, FC7 and FC8). In the network, Conv1, Conv2 and Conv5 layers are followed by the pooling operations (Pool1, Pool2 and Pool5) with the overlapping pooling units. Altogether the network comprises $650,000$ neurons.

Given a pre-trained network, the weights of an arbitrary convolutional layer, common choices are Conv3 and Conv5, can be used to extract the feature maps for the query image. These feature maps can be unrolled as a single descriptor. The dimensionality of the descriptor depends on the layer leveraged for feature extraction. The shallow layers in the network are generally very high-dimensional. For instance, unfolding feature maps of the Conv3 layer into a single vector yields a $13 \times 13 \times 384 = 64,896$ dimensional descriptor. Nevertheless, it is worth pointing out here that the first convolutional layer in the hierarchy learns the preliminary responses, such as opponent colors, oriented edges, etc., whereas the deeper layers learn more complex and generic features, and so they represent the abstract information.

2.7 Biology of Visual Perception, Learning and Navigation

Biophysical systems (including humans) possess tremendous capability of perceiving, interacting, learning and navigating in diverse environments. These cognitive processes evolve constantly through experiences, while their coordination and governance takes place in brain. Early research in cognitive psychology is guided by behaviorism – inferring the mental processes by studying the behavior of mammals, typically human, cat, rodent, or primate, in relation to stimuli. However, recent technological advancements, such as electroencephalogram (EEG) and functional magnetic resonance imaging (fMRI), have brought about the new methods for measuring the responses of various brain regions (Goldstein 2011).

Our sensory organs respond to stimuli in form of electrical signals. These signals are transferred to brain for further processing and analysis. The very first area where the signals arrive from a sensory organ is known as the primary area of that sense, located in cerebral cortex of the brain[5]. The cerebral cortex is divided roughly in four lobes, namely: frontal lobe, occipital lobe, parietal lobe and temporal lobe, as shown in Figure 2.17. The frontal lobe is responsible for coordinating the information received from different senses, which includes planing, problem solving, concentration, etc. On the other hand, the primary areas of visual cortex (e.g. V1−V4) are found in the occipital lobe. Early vision deals with light adaptation, opponent color processing, on/off center-surround edge-like responses at multiple scales and orientations, temporal decorrelation (motion), depth, etc. This information is held for few hundred milliseconds in iconic memory of visual perception. From visual cortex, the information flows into posterior parietal cortex, known as the dorsal stream. It is where the shape of object is formed using motion cues. Besides this, the parietal lobe has important role in pain and touch sensation. It also partly takes care of associating information from several senses, e.g., perceiving and acting to grasp a cup. The visual information also travels down to temporal cortex, i.e., the ventral stream. It is where the recognition and identification tasks occur. Indeed, temporal lobe is the chief area for auditory signal processing and recognition tasks. As it contains the hippocampus, surrounded by pre- and postrhinal cortices, memory formation for recognition and spatial navigation takes place in this region, follow the read at University of Bristol (2011).

In the scope of this research, we explore selected areas in occipital and temporal lobes at the functional level, as they are essential to visual perception, recognition and navigation tasks. Here, one should note that the functional organization of the brain

[5]The cerebral cortex is the thin layer of neural tissues making up the anterior part of the brain, called cerebrum (Goldstein 2014).

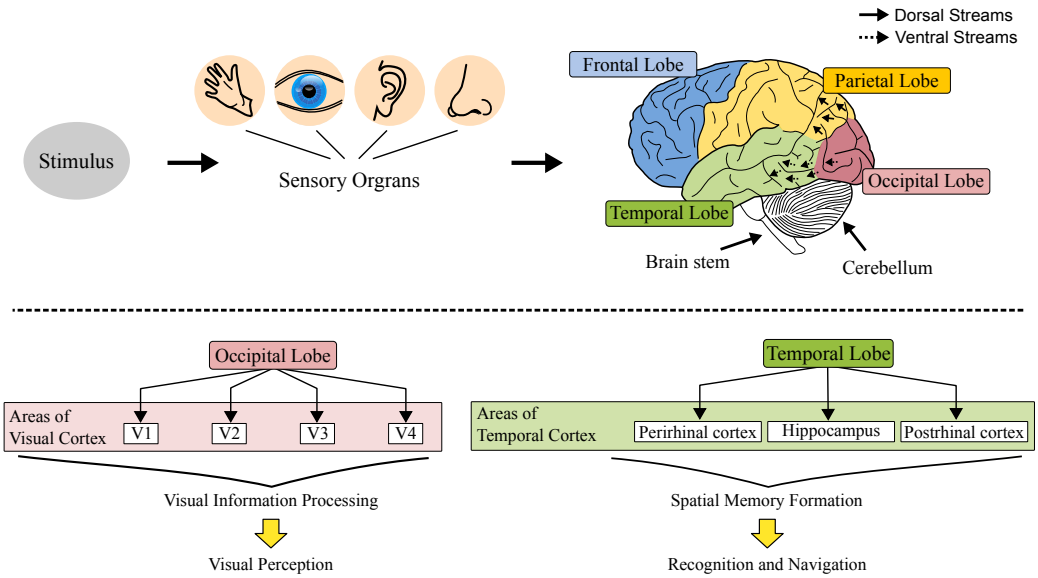


Figure 2.17: Signals arriving at the brain from a sensory organ are processed in the primary area of that sense. For instance, visual information from eyes is perceived in primary areas of visual cortex (V1−V4), whereas the memory formation and recognition tasks are done in temporal lobe, that is the area where the hippocampus is located.

regions does not imply that the information generation and retrieval is localized to those areas. Indeed, these regions work in coordination and the modern-day neuroscience has to discover yet the working principals in which these regions coordinate. Hence, the bio-inspired technical systems are not strictly biologically plausible. Besides this, the researchers are divided among the computational models of the specific cells in the brain, see for instance (Arleo and Gerstner 2000; Milford and Wyeth 2008). Nonetheless, the present studies give a fundamental insight into the mental processes, such as *visual perception*, *learning* and *navigation*. In the following discussion, we briefly touch upon these processes, while the later chapters translate these cognitive aspects to the technical system for learning the natural responses to a scene and persistent navigation.

2.7.1 Visual Perception of Space

Visual perception can be defined as processing and analysis of visual sensory data that our eyes send to the visual cortex. Our eyes are complex sensory organs that receive stimulus from the environment in form of light. When the light entering the retina hits the photoreceptor cells (rod and cones), the sensation of light energy is transduced into signals that travel (via optic nerve) to the visual cortex for processing the visual information at different levels of cognition. There exist two competing theories regarding the process in which visual information is processed in our brain. For a group of researchers (Gibson 1966; Marr and Hildreth 1980), visual perception

develops progressively from the finer details to the complex analysis of input. That is, all the perceptual information comes from eyes (bottom-up processing). Others, for instance (Gregory 1997), suggest that our perception is rather guided by the knowl–edge that we posses beforehand. Hence, visual perception and feature integration is a top-down mechanism of information processing. Said another way, bottom-up theory refers to the data-driven processing of visual stimulus, while the top-down theory argues the influence of contextual information in perception. For a thorough overview, the interested reader is referred to (Buschman and Miller 2007; Connor et al. 2004; Goldstein 2011)

Modern-day movie trailers switch rapidly between different scenes and contexts, but we are still able to understand the events and semantic category of each scene, without the fine-grained analysis of every single detail in images (Goldstein 2014; Potter 1976). This suggests that humans are able to capture the meaning from pictures within a small fraction of a second, referred to as a *gist of the scene* (Oliva and Torralba 2006). To demonstrate this phenomenon, (Potter 1976) presented a sequence of rapidly changing images to the subjects and observed that the partici–pants successfully indicated the pictures they saw. In another experiment, (Fei-Fei et al. 2007) studied that observers were able to report the meaning of a scene within a quarter of a second (i.e. less than 250 ms). They further studied that computing the gist of a scene may happen in less than 100 ms, if the exposure to a new scene or eye fixation has a lag of few milliseconds. Indeed, when the images flash for less than 100 ms, an observer will perceive only few large blobs/objects describing the perceptual layout of the surroundings. Whereas, increasing the time to 500 ms results in collecting more details from the image. Hence, first the gist is perceived from a scene, followed by the detection of further details and smaller objects.

Others (Biederman 1995; Oliva and Torralba 2006; Schyns and Oliva 1994) extended the experimental study to investigate the role of spatial arrangement of objects for scene categorization. To this end, they presented the blurry scenes to the participants and asked them to assign a semantic category to images based on relationships among blobs, see an example in Figure 2.18. Looking at the blurred image in Figure 2.18a gives an impression of the scene as a roadway with buildings alongside. However, in the original image, i.e., Figure 2.18b, one can observe that the top part of the image, which one interprets as buildings, actually comprises the cabinet and refrigerator in the kitchen. This misinterpretation of the blurred image[6] does not refer to a defect in a human vision system. It rather strengthens the fact that spatial layout information constrains the objects' identification. Following the bottom-up theory of perception, (Oliva and Torralba 2001) extracted the gist from

[6]The frequency content in the image was smoothed using Gaussian filter (Figure 2.18c). The smooth–ness level σ is given by $f_c/\sqrt{log(2)}$ (Oliva and Torralba 2006), where f_c is the number of cycles per image.

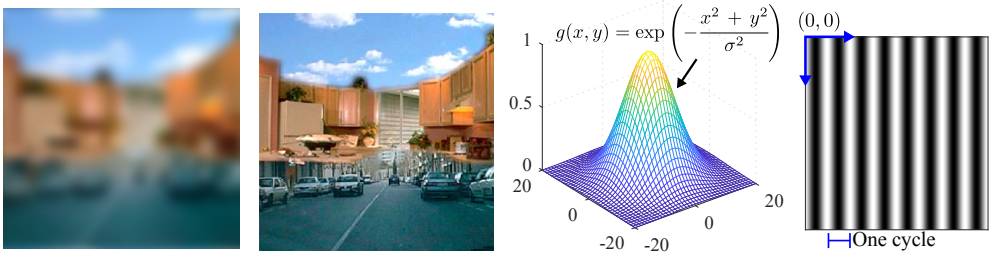


Figure 2.18: Effect of coarse visual details on scene categorization. (a) Blurred image retains the spatial frequency of eight cycles per image. Looking at the image gives an interpretation of buildings alongside the roadway. (b) The original image (Oliva and Torralba 2006) reveals the misinterpretation of cabinets as buildings. (c) Gaussian filter applied to (b) to produce (a). (d) A cycle in an image is a transition from one intensity level to another and again back to original one. The grating image, i.e., a repeated sequence of dark and light bars, illustrates this concept (vertical spatial frequency of eight cycles/image).

images by describing images in term of perceptual attributes, such as a degree of naturalness, openness, expansion, etc., as shown in Figure 2.19. They suggest that these properties are shared among places and are thus helpful for characterizing images, without an explicit identification of the objects. The scene of buildings and man-made structures, here Figure 2.19a and Figure 2.19b, have horizontally dominant or cross-shaped frequency distribution on the power spectrum. Natural scenes have a vertically dominant (Figure 2.19c) or an isotropic (Figure 2.19e) power spectrum, while the closed environments (scenes with boundary elements) have an oblique (Figure 2.19d) or a circular (Figure 2.19e) power spectrum.

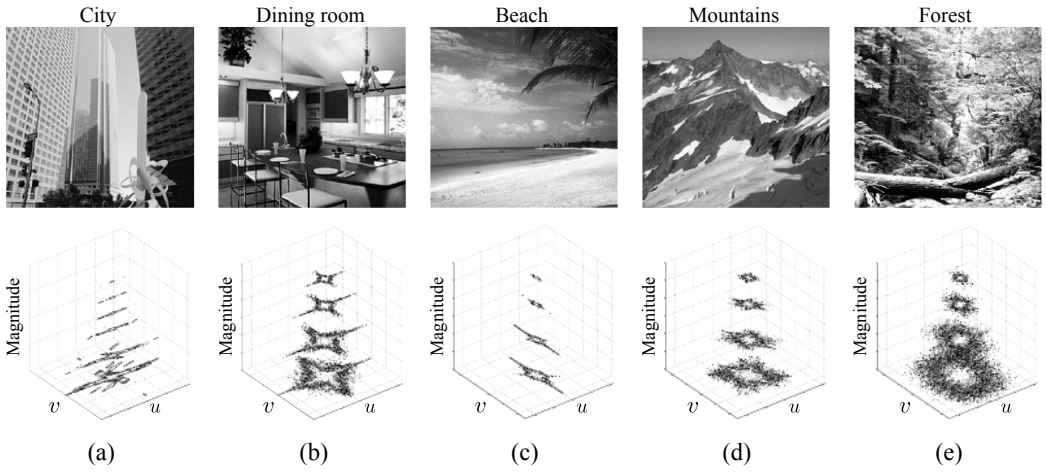


Figure 2.19: Power spectrum of the example images, modified on (Oliva et al. 1999, Figure 1). The magnitude is in logarithmic scale, and the variables u and v refer to the horizontal and vertical spatial frequencies. (a) horizontal (b) cross-shaped (c) vertical (d) oblique and (e) isotropic power spectrum. Note that the shape of the power spectrum varies based on the semantic category of the scene.

2.7.2 Learning from Stimuli: Self-organizing Maps (SOM)

The patterns of input signals that brain receives from sensory organs are learned by specialized types of cells (neurons). These cells are found in various regions of the brain, including primary areas of visual cortex and the hippocampus. They show competitive responses to learn from stimuli and organize themselves in form of a map, also called as a cortical feature map (Mikkulainen 2005). The neurons that get tuned to learn a particular type of feature exhibit high response to represent that input. An artificial neural network that closely models the self-organizing behavior of biological cells in cortical regions of the brain is known as Kohnen's Self-Organizing Map (SOM) (Kohonen 1990). Similar to the biological neurons, a SOM preserves the topology of the input space. That is, the data points that lie close to each other in the input space, they remain close in the learned (coarse) representation. The architecture of a SOM comprises an input layer and a Kohonen layer. Every input vector is fully connected to the neurons in a one- or two-dimensional lattice of neurons in Kohonen layer. Figure 2.20 illustrates a SOM network for a one-dimensional lattice. Each neuron in the lattice has a unique index number i. Training a SOM generally involves three steps: network initialization, searching a winner neuron and weights adjustment.

Network Initialization

The network initialization requires the decision of the network size, i.e., the number of neurons N in the network. Selecting a suitable network size is critical to the quality of maps learned by the network, as too few neurons underfit the input space

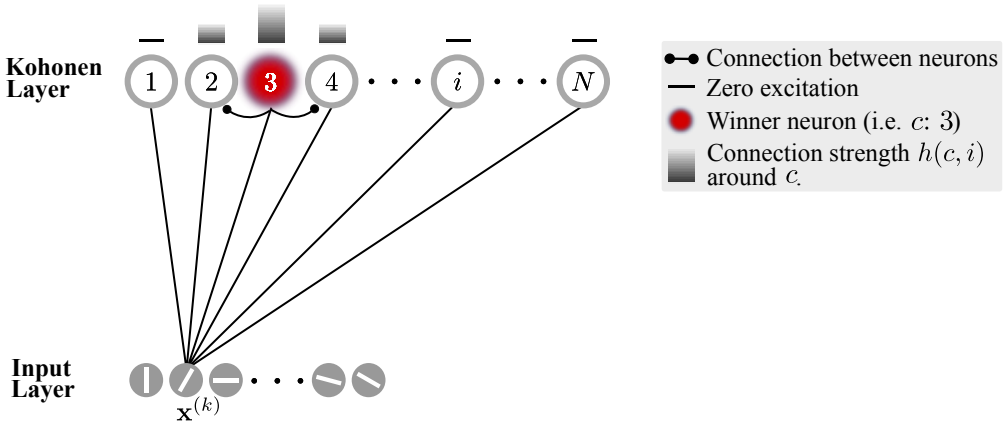


Figure 2.20: A SOM example for a one-dimensional case (Kazmi and Mertsching 2019a): nearby neurons of the winner (in lattice) are referred to as its topological neighbors. The neighbors of the winner neuron share the excitatory connec–tions to learn the input, where the strength of excitation depends on their lateral distances from the winner neuron (see text for details).

and hence the network will never converge[7] to a stable pattern of activity. Whereas, too many neurons overfit the input. After deciding on the network size, the next step is initializing the weights $\mathbf{w}_i$ $(1 \leq i \leq N)$ of the neurons. There exist several weight initialization schemes, such as assigning random weights or sampling arbitrary data points as the weight vectors.

In the subsequent discussion, we assign the time superscript t to the weight vectors $\mathbf{w}_i^{(t)}$ to refer to the weights of the neuron i at a time instant t. The weights of all the neurons are denoted by a $L \times N$-dimensional matrix $\mathbf{W}^{(t)}$. Hence, the network's state $\mathcal{S}^{(t)}$ at a time t is simply described by the weight matrix and a set of network parameters $\Phi^{(t)}$ at that time, i.e., $\mathcal{S}^{(t)} = (\mathbf{W}^{(t)}, \Phi^{(t)})$.

Searching Winner Neuron

Every time an input $\mathbf{x}_j^{(t)}$ is fetched from the training data, its distance to weight vectors of all the neurons is computed. There are various distance measures, such as Euclidean, Manhattan, Mahalanobis, etc., but the choice of the distance metric depends on the feature space and the nature of application. For instance, for recog–nition tasks, typically a sum of squared residuals is used. The neuron c with the minimum distance from the input is said to be the best representative of the input vector and is deemed a *winner* or *best matching neuron*:

$$c = \arg\min_i \left\| \mathbf{x}_j^{(t)} - \mathbf{w}_i^{(t)} \right\|^2$$

where j indexes over the training set of M examples.

Weight Adjustment

Neurons learn from the inputs in a training epoch. The definition of a training epoch differs with regard to the weight update strategy, namely batch or online. In a batch version, the training examples are available beforehand, thus the training data is traversed multiple times to adjust the weight vectors. In this case, one complete traversal through all the training examples is referred to as a training epoch. On the contrary, an online learning is stochastic. This means that the network sees the input only once and after adapting the weights the input is discarded. In case of an online learning, feeding a single training example to the network is a training epoch. Note that for a batch learning, inputs are picked randomly instead of sequential retrieval.

[7] A network is said to be converged if the sum of mean square error of each input $\mathbf{x}_j \in \mathcal{R}^L$ from the representative neuron c is within an acceptable threshold.

Such an approach facilitates the network learning hidden patterns in the data and preventing the sensitivity to the order of input presentation.

An obvious question that could arise here is that which neurons should adapt their weights for the given input. Two commonly used methods are: *winner-takes-all* and *winner-takes-most* mechanisms. The former adapts the weight vector of only the winner neuron, which is given by the following weight update rule:

$$\mathbf{w}_i^{(t+1)} = \begin{cases} \mathbf{w}_i^{(t)} + \alpha\left(\mathbf{x}_j^{(t)} - \mathbf{w}_i^{(t)}\right) & i = c \\ \mathbf{w}_i^{(t)} & i \neq c \end{cases} \tag{2.13}$$

where the learning rate $0 < \alpha < 1$ can be a small constant value, which gradually drags the neurons towards input vectors. Alternatively, it may be a decaying function of training epochs (iterations), e.g., $\alpha = \alpha_0(\alpha_f/\alpha_0)^{\tau/T}$, as chosen in (Fritzke 1997). The parameters α_0 and α_f are the suitable initial and final learning rates set by the user; T specifies the total number of training iterations and τ is the current iteration number. An optimal learning factor depends on the application and design decision of SOM (Kohonen 1997). Note that the winner-takes-all rule is a hard competitive learning method, because only the winner is selected to learn the input. On the other hand, winner-takes-most rule is a soft competitive learning approach, as the weights of the winner as well as its neighbors are adapted:

$$\mathbf{w}_i^{(t+1)} = \mathbf{w}_i^{(t)} + \alpha\, h\Big(c, i\Big)\left(\mathbf{x}_j^{(t)} - \mathbf{w}_i^{(t)}\right) \tag{2.14}$$

As a result of aforementioned update rule, neurons in the network form a cluster of activity around the winner c to learn the inputs. The excitation strength of a particular neuron i is given by the kernel $h(c, i)$. Hence, the distant neurons from the winner have zero excitation, while the nearby neurons get high excitation. Because the lateral interactions among the biological neurons are bell shaped, Kohonen (1997) has suggested a scalar Gaussian function:

$$h\Big(c, i\Big) = \exp\left(-\frac{d_e(c, i)^2}{\sigma^2}\right) \tag{2.15}$$

Here, the numerator $d_e(c, i)$ refers to the Euclidean distance between the winner c and the neuron i on the lattice. The kernel bandwidth σ defines the neighborhood size around the winner, which decreases monotonically as a function of training iteration $\sigma = \sigma_0 * e^{-\sigma_0\,\tau/T}$. The constant σ_0 is the initial neighborhood radius around the winner. Note that this definition of σ is only valid for batch learning. For online learning, setting T in advance is impractical, as training data is not available ahead of time. Algorithm 1 summarizes the steps for training SOM in a batch manner.

Algorithm 1 General steps for training SOM (batch learning)

$\alpha \leftarrow \alpha_0,\ \sigma \leftarrow \sigma_0$ ▷ Initial parameter values
$\mathbf{W}^{(0)} \leftarrow$ initialize_weights()
for $\tau \leftarrow 0$ to T **do** ▷ Repeat until T training epochs
    while more training examples **do**
        $\mathbf{x}_j^{(t)} \leftarrow$ retrieve_training_example()
        $c \leftarrow$ search_winner_neuron$(\mathbf{W}^{(t)}, \mathbf{x}_j^{(t)})$
        $\mathbf{W}^{(t+1)} \leftarrow$ adjust_weights$(\mathbf{W}^{(t)}, \mathbf{x}_j^{(t)}, \Phi^{(t)})$
    end while
    $\alpha \leftarrow \alpha_0(\alpha_f/\alpha_0)^{\tau/T}$ ▷ Learning decay
    $\sigma \leftarrow \sigma_0 * e^{-\sigma_0 \tau/T}$ ▷ Shrink the neighborhood size
end for

2.7.3 Biological Navigation

The structures inside the temporal cortex, i.e., hippocampus and the adjacent regions, such as the peri- and postrhinal cortices, are critical to visual recognition and spatial memory for navigation. Neurons in these regions operate as a network to exchange the information. Certainly, the brain of the rats is one of the biological systems that has been extensively studied by the scientists to get a deeper insight into the visual recognition and memory formation mechanisms (Arleo and Gerstner 2000; Barrera and Weitzenfeld 2008; Wyeth and Milford 2009; Xiang and Brown 1998). In the following discussion, we introduce the types of cells that are responsible for recognition and spatial memory.

Visual Recognition Memory

The neurons that are specialized to perform the visual recognition tasks, e.g., rec–ognizing objects in the range of visual field, are found in the perirhinal cortex – a cortical region alongside the hippocampus. Certain neurons (novelty neurons) in this region respond to novel inputs, for instance, seeing an image that was never seen before. The response of such neurons sustain for some time. These neurons respond less to the familiar inputs. Other neurons, called as familiarity neurons, respond to the familiar stimulus. They lose the sensitivity to the familiar input on observing it multiple times. Yet, there exist another type of cells (recency neurons), which respond to the inputs from the recent past. At start, they show high activity to the recently observed stimulus, while their activity drops off when the input is seen again. Nevertheless, these neurons quickly recover their responsiveness to the stimulus (Xiang and Brown 1998).

The process of visual recognition is not confined to the novelty/familiarity neurons. It also depends on the spatio-temporal information of the objects, e.g., spatial ordering

of the objects or the time order of seeing these objects. Such an information is maintained in hippocampus. The neurons in perirhinal cortex receive information from different cortical regions, including hippocampus and entorhinal cortex, in form of synapses. The integration of the incoming signals is done in the cell body and the response travels down the axon to form the recognition memory. This is the reason why, we recognize the dislocation of the familiar objects in the room, such as when a table is put out of its previous location. The communication between these two regions govern the tasks such as recognizing familiar objects or faces, associating sensory information with memory, and understanding the context of sensory experiences. The connections allow for the integration of sensory input and the retrieval of relevant memory information, contributing to the ability of perceiving the surroundings.

Spatial Memory

(O'Keefe 1979) studied that the neurons in hippocampus and medial entorhinal cortex show high firing rate, when a free moving rat changes its position in the environment. These cells were thus named as *place cells*. While there exists ample literature investigating the role of various types of cells in spatial memory formation (Best et al. 2001; Hartley et al. 2013; Wyeth and Milford 2009), place cells build inner map of the physical space in brain. These cells are strongly correlated with the absolute location of a rodent and are invariant to the heading direction of the animal. A place cell shows maximum activity for a seen place, which drops gradually as a rodent moves away from the place. This results in the formation of spatial memory, i.e., the map of the environment.

Figure 2.21 illustrates the concept of a place cell in a circular space. Numbers in the figure refer to the place cells at three different locations. A place cell has one (or more) associated place fields, where each place field maps a different location in the space. This re-mapping of the place cell to different parts of the space enhances animal's ability to represent more places in the environment. The position of a place field is set relative to spatial cues. Thus, the recognition of a familiar object makes it easier to locate the other objects in the space with reference to that object. This suggests that the spatial cues act as anchor points, hence moving the spatial cues will relocate the place field by that amount. The place fields can be

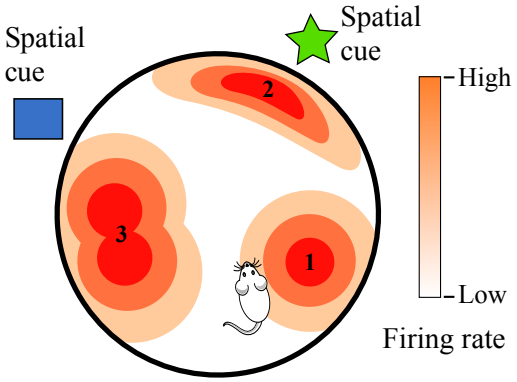


Figure 2.21: Example: Place cells in a circular space.

circular or any other shape depending on the region they map in the space. A place field can have many sub-fields.

Other cells that are involved in spatial memory formation include: *head-direction* and *grid* cells. Head-direction cells are found in several regions nearby hippocampus, such as post-subiculum, retrosplenial cortex, and some regions of thalamus (Taube et al. 1990). These cells respond maximally when the rat faces a particular direction in the environment. On the other hand, grid cells are discovered in entorhinal cortex (Hafting et al. 2005). They exhibit place cell like properties, but they have multiple firing fields, which repeats in a hexagonal pattern. A single grid cell fires when a rat is located at any vertex of a tessellating hexagonal pattern across the environment. At a superficial level, some grid cells show a tessellation in place and no response for the direction. The cells in deeper layers exhibit characteristics similar to the head-direction cells, while the deepest layers show conjunctive properties, for instance, when rat is at a certain location and has a specific orientation. Altogether, the place, head-direction and grid cells form the positioning system in the brain. It is, however, noteworthy that the brain maps are not strictly geometric, they are rather topological (Wyeth and Milford 2009).

3 Related Works

Visual loop-closure detection is a widely known map correction and re-localization tool in any visual SLAM system. Recent research in this regard has led to the discovery of heterogeneous algorithms: some scholars craft features that are less sensitive to illumination conditions (Carlevaris-Bianco and Eustice 2014; Kawewong et al. 2011; Lategahn et al. 2013) and appearance changes (Krajník et al. 2017). Others (Chen et al. 2014; Sünderhauf et al. 2015a) benefit from the pre-trained convolutional neural networks (CNN), e.g., AlexNet (Krizhevsky et al. 2012) or Overfeat (Sermanet et al. 2014), to extract the condition-invariant features. There exist methods (McManus et al. 2015; Ranganathan et al. 2013) that model the appearance of locations from repeated observations of places at multiple times; on the other hand, the works, such as (Lowry and Milford 2016a), tend to learn linear transformations between different appearances of the scenes. It is also common among a group of researchers to exploit off-the-shelf features for learning the visual vocabulary (Cummins and Newman 2011; Garcia-Fidalgo and Ortiz 2018; Johns and Yang 2014; Murillo et al. 2013) or matching sequences (Chen et al. 2015; Milford and Wyeth 2012; Naseer et al. 2017b; Pepperell et al. 2014). For a walk-through of the state-of-the-art place recognition methods, including their fusion with the geometric information, through other sensory modalities, we refer to (Cattaneo et al. 2021; Garcia-Fidalgo and Ortiz 2015; Lowry et al. 2016b; Wang et al. 2021; Williams et al. 2009; Xu et al. 2021); whereas very recent advancements are summarized in (Schubert et al. 2023; Tsintotas et al. 2022). Following discussion reviews the state-of-the-art loop-closure detection methods with focus on their learning and recognition methodologies.

3.1 Vocabulary-based Place Recognition

The vocabulary-based methods model the visual appearance of a scene as a histogram of the visual words observed in an image. A detailed procedure to generate the visual words is already explained in Section 2.6.2. FAB-MAP 2.0 (Cummins and Newman 2011) is the most popular vocabulary-based place recognition framework. It learns the visual vocabulary of SURF features and exploits Chow Liu tree for capturing the dependencies in co-occurrence of visual words. Such a scheme has enabled the authors to approximate the joint probabilities optimally in a recursive Bayes

formulation. Murillo et al. (2013) made use of a k-d tree to learn the gist features based vocabulary. This has improved the descriptor matching efficiency in their case. Finally, the problem of loop-closure detection is formulated using discrete Bayes filter and the maximum of the posterior probability is used to detect the visited locations. Others, e.g., (Mur-Artal and Tardós 2014), trained a tree structure for a large offline vocabulary of binary features, i.e., ORB. Their approach builds upon the DBoW2 (Gálvez-López and Tardós 2012), which trains a vocabulary of BRIEF features. They introduced a novel indexing approach to efficiently find the feature correspondences for geometric verification. The nearest neighbor (NN) approach was used for matching the locations.

The need of an offline training phase, degrades the performance of the visual vocabulary-based approaches in previously unseen environments, as stated in (Chen et al. 2015; Garcia-Fidalgo and Ortiz 2017; Glover et al. 2010; Pepperell et al. 2014; Yeh et al. 2007). For online vocabulary training, Nicosevici and Garcia (2012) proposed a modified agglomerative clustering algorithm. An essential part of their method is the vocabulary update scheme which needs a little more than a second for each vocabulary update. Others, e.g., (Angeli et al. 2008; Filliat 2007), followed an incremental NN classifier approach for online vocabulary generation. However, these methods are tested only for small environments. In order to decide on useful features for online vocabulary learning, Khan and Wollherr (2015) rely on the tracking of the BRISK features over consecutive frames. To retrieve loop locations, they introduced a specialized indexing scheme and maximized a likelihood function on the basis of occurrence frequency of visual words in images. Finally, a consistency test is followed to filter out the inconsistent loop-closure hypotheses. In the same vein, Zhang et al. (2016) has trained the binary codewords via Linear Discriminant Analysis (LDA) and extended (Khan and Wollherr 2015) for detecting loops.

Recently, Garcia-Fidalgo and Ortiz (2017) introduced a hierarchical procedure for loop-closure detection. Their algorithm initially selects the candidate loop locations by matching the global descriptor (PHOG) of a query image with the incrementally created map of locations. After the initial selection, the local binary features (LDB) and the visual vocabulary of the binary words are utilized to retrieve the most likely matches, which finally follow the geometric consistency check to find a potential loop-closure location. More recently, the authors have proposed another approach, which they dubbed as iBoW-LCD (Garcia-Fidalgo and Ortiz 2018). This algorithm uses the concept of dynamic islands for grouping images subject to their spatio-temporal similarity. To speed-up the indexing, the search space formed by the ORB features has been clustered in a tree structure. The experimental results demonstrate the superiority of the approach compared to (Garcia-Fidalgo and Ortiz 2017; Khan and Wollherr 2015; Zhang et al. 2016). Although the aforementioned works have made ample use of binary features for the purpose of efficient vocabulary

update and location matching, the binary feature suffer from repeatability issue in case of low image resolution or abrupt brightness changes, which eventually deteriorates the place recognition performance.

3.2 Matching Sequences for Place Recognition

Sequence-based approaches focus on matching sequence of images rather than comparing single images. To this end, the similarity score between the source and target image sequences is minimized. The idea of sequence matching was pioneered by Milford and Wyeth in their work for place recognition under varying weather conditions (Milford and Wyeth 2012), known as SeqSLAM. To achieve localization in the space of visual appearance, the SeqSLAM algorithm computes the difference matrix, which stores distances between the source image sequence and the database images. Assuming that the vehicle moves with the constant velocity, the linear trajectories are generated over the difference matrix and the minimum scoring trajectory, by the virtue of dynamic time warping (DTW), is regarded as a previously visited route.

Pepperell et al. (2014) introduced the variable-offset matching scheme to strengthen the SeqSLAM's performance in presence of the variable velocities. As the sky region in images does not contribute to the localization information, the authors opt for the sky blackening transforms which nullifies the sky in images. Moreover, the use of odometry data has enabled them to effectively linearize the trajectories, as the templates are learned for a constant change in distance. Some works, e.g., (Jacobson et al. 2015), focus on automatic parameters' calibration to improve the SeqSLAM's performance in unknown environments. Naseer et al. (2017b) introduced a novel state transition model and cast the sequence matching problem in terms of Bayes filter, which outperforms their network flow method (Naseer et al. 2015). Chen et al. (2015) adopted the large margin nearest neighbor (LMNN) algorithm to improve the place recognition performance of the SeqSLAM. Lately, Siam and Zhang (2017) presented Fast SeqSLAM, which uses an approximate nearest neighbor approach and the strategic search scheme to gain significant speed-ups in sequence matching.

3.3 Learning Spatial Representation for Place Recognition

Artac et al. (2002) incrementally trained the eigenspace model of the environment using principal component analysis (PCA). For each query image, the reconstruction

error and the projection coefficients are determined in the current eigen space. To account for the changes from new images, the eigenvalue problem is solved and the model is updated accordingly. Luo et al. (2007) globally represent images using composed receptive field histograms (CRFH) and employed an incremental support vector machine (SVM) for learning the representation of a small indoor environment. Others, e.g., (Han et al. 2017) modeled the appearance of a location using various image features. For each feature modality, they learned the weight via regularized optimization scheme. Hence, the approach is dubbed as the shared representative appearance learning (SRAL). Later, the similarity between the query and database images is computed by exploiting the learned weights.

For learning a sparse visual representation of the test environment, Kosecka et al. (2003) made use of learning vector quantization (LVQ) technique. Their method involves two steps: in an offline training phase the histogram of image gradients is computed followed by non-maximum suppression and analysis of the connected components. The code vectors of the input space are learned using the steepest gradient-descent strategy. Given a trained codebook, the ratio of the chi-square distances between the query image descriptor and the two nearest code vectors is computed and the place is flagged as known, if the ratio surpasses the pre-set threshold. (Kawewong et al. 2011) incrementally learned the model of places using position invariant robust features (PIRF). A PIRF descriptor is an average over the SIFT features that do not significantly change their position under successive camera motions. For detecting a loop-closure, the normalized score is calculated following a set of similarity assessment criteria. The performance bottleneck of this technique is the temporal window size for computing the PIRF.

To learn condition-invariant attributes of the environment, a matrix factorization approach, i.e., singular value decomposition (SVD), has been adopted in (Lowry and Milford 2016a). The authors retained the least significant eigenvectors of the right-unitary matrix to extract the condition-invariant statistics from the images. In the same paper, the authors have also touched upon the linear regression technique to predict the appearance changes at multiple times of the day. McManus et al. (2015) deployed a pre-trained bank of SVMs to recognize places in extreme lighting conditions. For each query image, the algorithm extracts HOG features from sampled patches of variable length, called distinctive scene signatures. The features are fed to the pre-trained SVM classifiers that fire independently for the specific visual element in the scene. Thereby, the classifiers crudely define the location of the vehicle in an environment at a low geometric precision.

Gehrig et al. (2017) proposed a probabilistic framework that relies on a sparse feature map of the environment generated by the pose-graph algorithm (Grisetti et al. 2010a). Given a query image, a kNN (k-nearest neighbor) search is performed

in the projected space of BRISK descriptors and a vote count of the best matching vertices is aggregated. Lastly, a probabilistic score is assigned to each vertex using the binomial distribution. If a highest scoring vertex passes the geometric verification step, the query frame is regarded as a potential loop-closure. Some authors, such as (Sünderhauf and Protzel 2011), computed the binary holistic representation of the scene using BRIEF descriptor, named as BRIEF-Gist. For this purpose, the image is down-sampled and the descriptor is extracted around the image center. The procedure of the loop-closure detection is thus simplified to descriptor matching. Others, e.g., (Wu et al. 2014), processed the raw images to the binary form and defined mutual information criterion as a similarity measure between the image pair. To demonstrate place recognition for large viewpoint changes, Torii et al. (2015) represented the images with densely sampled SIFT descriptors and synthesized the virtual views of query locations using Google street-view and coarse depth maps of the locations. For efficient retrieval and compact storage, the image description is encoded. A nearest neighbor approach is applied to search the query in the database. Nevertheless, such a scheme is only applicable to the places which are mapped in Google street-view.

3.4 Bio-inspired Paradigms

The motivation of bio-inspired methods is to mimic the biological processes underpinning navigation performed in mammalians' brain (including humans). For instance, Arleo and Gerstner (2000) studied the hippocampus activity in a rat's brain and presented the simplified model of the place and head-direction cells using continuous attractor neural network (CANN) – a type of self-organizing recurrent neural network with lateral connections between the neurons. The model is tested in a small arena with bar-coded walls. The network shows high activity upon seeing a familiar input. An unfamiliar visual input induces activity in another cluster of neural units, which gradually disappears if the motion and visual cues do not support the new cluster of neurons.

Hafner (2000) used a self-organizing neural network to create the representation of an environment with neurons exhibiting the place cells like behavior. In order to augment the map with metric awareness, a physical force model is introduced, which describes the spatial distances and orientation between the neurons. As the network was initialized with random weights, the algorithm yields different map each time the tests are performed in a simulated or real-environment. Vassallo et al. (2002) proposed a method of learning visual scenes together with the motor commands executed between the scenes. The system learns a visual scene if it varies by a significant amount. The plus point of learning this association is that it supports

the bi-directional traversing for the learned environment. However, weakness of this approach lies in the fact that unknown, cluttered and dynamic environments are highly uncertain and so the motion is learned between two visual scenes. A similar approach was developed by Hamze and Clark (2001), however, it used Kohonen's self-organizing neural network to associate camera images with motor commands.

Some authors, e.g., (Siagian and Itti 2009), trained a neural network to model the appearance of places based on gist features, salient regions and SIFT features. For determining the most likely location, a modified version of the Monte Carlo approach is employed. The system was tested in a small outdoor environment. Rebai et al. (2013) exploited the fuzzy adaptive resonance theory (Fuzzy ART) and emulated the behavior of view cells in visual cortex of primates to learn visual representations of the environments. RatSLAM (Milford and Wyeth 2008) is by far the popular and pioneer bio-inspired robotic mapping approach. It has been demonstrated to map a large suburb of Brisbane. It draws upon (Arleo and Gerstner 2000) with some modifications in the CANN model for the head-direction and place cells. The system learns the associations between visual appearance of the scene and the pose of the robot, which are maintained in local view and pose cells, respectively. The path-integration information from odometry is utilized to shift activity in the pose cells network. The errors accumulated due to path-integration are calibrated by visual input from local view cells, while simultaneously recalling the prior associations learned between the local view and pose cells. Unfortunately, the system's performance heavily relies on the parameter settings and the size of a pose cells network.

Glover et al. (2010) fused the FAB-MAP algorithm with RatSLAM to boost the place recognition performance of RatSLAM and relax the parameter tuning overhead. Their experiments show that the offline vocabulary training is sensitive to illumination conditions and it did not help them removing the system's dependency on parameters. In the same vein, Kazmi and Mertsching (2016a) replaced the front-end module of the RatSLAM with a modified GSOM-based network for learning and recognizing places. The proposed model imitates the concept of input recency and familiarity in the perirhinal cortex. Their experiments report significant improvements in the place recognition performance and thus assisted RatSLAM to correct the drifts in the topological map – see Chapter 4 for more details.

3.5 CNN-based Place Recognition

CNNs principally earned fame because of their tremendous performance for object recognition tasks (Krizhevsky et al. 2012). Thereon, they have been widely adopted

in other domains of computer vision, including place recognition (Chen et al. 2014; Sünderhauf et al. 2015b). An in-depth comparison of various CNN architectures, including the computation power needed per network, execution time, accuracy, etc., is presented in (Canziani et al. 2016). More recently, Ferrarini et al. (2022) proposed a binary neural network for efficient place recognition. In general, the literature on leveraging CNNs for place recognition tasks can be categorized in three main methodologies.

3.5.1 Leveraging CNN as a Feature Descriptor

Chen et al. (2014) pioneered the use a CNN for place recognition. To this end, the authors utilized a pre-trained AlexNet (Krizhevsky et al. 2012) and extracted the responses of the network layers as whole-image descriptors. Others, Sünderhauf et al. (2015b) and Neubert and Protzel (2015), exploited the `Conv3` layer of a pre-trained AlexNet and VGG-M, respectively, to extract the filter responses over the proposal image regions. As the network models were trained on objects' database, they lack the features needed for place recognition. Zhou et al. (2014) showed that the accuracy of the AlexNet for place recognition can be improved, if it is trained on a place-centric database, as the network learns the features that are essential to discriminate places than distinguishing objects.

Moreover, not all the layers in the network fit equally well to address the challenges of place recognition. According to an experimental study (Sünderhauf et al. 2015a), the shallower network layers, e.g., `Conv3`, contain finer features and so they are robust to appearance variations. By contrast, the features learned by the deeper layers (e.g. `FC6`) are robust to viewpoint variability.

3.5.2 Encoding CNN Output into a Feature Descriptor

Some authors (Chen et al. 2017; Garg and Milford 2021; Krizhevsky et al. 2012), opt to encode the feature maps of CNNs into a single vector using techniques, such as vector of locally aggregated descriptors (VLAD) (Jégou et al. 2010), Fischer vector (Jaakkola and Haussler 1999), etc., followed by various pooling operations. The pooling operations typically enhance the sparsity of the feature maps and thus help picking up important structures in images.

Instead of aggregating local features in a hardcore manner, Arandjelović et al. (2018) introduced VLAD as a trainable layer, called as NetVLAD. The layer is plugged to the `Conv5` of the cropped AlexNet and VGG-16 networks, allowing the network training in an *end-to-end* manner. This means that besides training NetVLAD layer,

the weights of the convolutional layers of the network are also adjusted. Unlike VLAD, which makes hard assignments to the residuals of an input from the cluster centers, their approach learns the soft assignments.

3.5.3 Using Semantics to Encode CNN Activations

Some researchers use 3D scene geometry and/or semantic segmentation (Naseer et al. 2017a; Schönberger et al. 2017) to guide the networks selecting important structures in images. In particular, Naseer et al. (2017a) builds upon a semantic segmentation network (Oliveira et al. 2016) to learn the structures in a scene that remain stable to appearance and viewpoint variations. As these structures are not always observable in images, they have also included the activation maps for the unsegmented version of image. Finally, a low-dimensional vector representation of the image is obtained using a dimensionality reduction technique. In order to compactly represent images, Yu et al. (2018) applied VLAD framework on the semantic edge information extracted from a specialized CNN.

In Table 3.1, we summarize the well-known place recognition approaches for ap–pear-ance-based mapping. It can be observed in the table that a mainstream of the existing approaches require an offline training, prior knowledge of the target environment, or an exhaustive parameter tuning before the system is functional. This degrades the applicability of such methods in previously unseen environments. On the other hand, the algorithms, such as deep networks, require extreme compu–tational resources. As one of the prime concerns of this work is to develop resource efficient place learning and recognition algorithms, without a prior training or knowledge of an environment, training a CNN for a feature extraction task is not the principal focus of this research.

Table 3.1: Well-known Place Recognition Approaches for Appearance-based Mapping (Adapted on (Kazmi and Mertsching 2019a))

Method	References	Features	Learning algorithm	Procedure	Recognition algorithm
Vocabulary-based methods†	Cummins and Newman (2011)	SURF	Approx. k-means; Chow Liu	Offline / unsupervised	Bayes filter
	Murillo et al. (2013)	Gist	PCA; k-d tree	"	"
	Gálvez-López and Tardós (2012)§	BRIEF	K-medians	"	NNS
	Mur-Artal and Tardós (2014)§	ORB	"	"	NNS
	Neubert et al. (2015)	Superpixels	Hierarchical k-means	"	Image synthesis & matching
	Nicosevici and Garcia (2012)	SURF	Agglomerative clustering	Online / unsupervised	NNS
	Garcia-Fidalgo and Ortiz (2018)§	ORB	Hierarchical trees	"	"
	Khan and Wollherr (2015)§	BRISK	Centroid based clustering	"	MLE
	Zhang et al. (2016)§	Binarized patches	LDA	"	"
	Angeli et al. (2008)§	SIFT, color histogram	Incremental NN	"	Bayes filter
	Garcia-Fidalgo and Ortiz (2014)	BRIEF	K-mediods	"	"
	Garcia-Fidalgo and Ortiz (2017)	PHOG, LDB	"	"	"
Sequence-based methods‡	Milford and Wyeth (2012)	Gray image	–	–	Dynamic time warping
	Chen et al. (2015)	Gray image, Gist, CNN	LMNN	Offline / semi-supervised	"
	Naseer et al. (2015, 2017b)	HOG, CNN	–	–	Network flow, Bayes filter
Learning spatial representation	Kosecka et al. (2003)	Gradient histogram	LVQ	Offline / supervised	NNS
	Han et al. (2017)	Gray image, HOG, LBP, CNN, Gist, SURF	Regularized optimization	Offline / unsupervised	"
	McManus et al. (2015)	Sampled patches	Bank of SVMs	"	SVM classifiers
	Artac et al. (2002)	Gray image	PCA	Online / unsupervised	NNS
	Luo et al. (2007)	CRFH, SIFT	Incremental SVM	Online / supervised	Support vectors
	Kawewong et al. (2011)§	PIRF	–	–	k-NN scoring
	Kazmi and Mertsching (2016b)	Gist	Growing SOM	Real-time / unsupervised	MAP
Bio-inspired methods¶	Hafner (2000)	Gray image	SOM	Online / unsupervised	NNS
	Milford and Wyeth (2008)	"	CANN	"	"
	Rebai et al. (2013)	Color histogram	Fuzzy ART	"	"
	Kazmi and Mertsching (2016a)	Gist	Growing SOM	"	"
CNN-based methods	Chen et al. (2014)	CNN	–	–	Sequential filter
	Sünderhauf et al. (2015b)	"	–	–	LSH with cosine similarity

† Vocabulary-based methods rely on a supplementary geometric validation test to reduce false detections. § Term frequency-inverse document frequency (Tf-idf) scheme is used, which assigns importance weights to the visual words. – No learning is performed by these works. CNN-based techniques exploit the pre-trained networks for feature extraction. ‡ These methods compute a difference (Milford and Wyeth 2012) or a similarity (Naseer et al. 2015; 2017b) matrix between the database and query images (descriptors); finding such a matrix is computationally intensive, as it depends on the number of database images and the pre-set sequence length. Moreover, sequence length parameter has to be adjusted according to environment (Jacobson et al. 2015). ¶ Popular bio-inspired systems, e.g., RatSLAM (Milford and Wyeth 2008), are accustomed to environment-related parameter tuning (Glover et al. 2010).

4 Bio-inspired Framework for On-the-fly Visual Learning

Biophysical systems, e.g., humans, navigate reliably through diverse environments without learning precise geometry of the physical space (Wyeth and Milford 2009). They rather use visual cues as a primary sensory input to associate places and thus preserve the topological structure of locations. First large-scale demonstration of this behavior is done in Brisbane, Australia, by the system named RatSLAM (Milford and Wyeth 2008). However, one of the primary shortcomings of this system is that it does not follow any learning procedure to build the visual representation of the environment. Rather the visual templates of the environment are created on ad-hoc basis. These visual templates are referred to as local view cells and they make up the front-end of the RatSLAM. Furthermore, the dependability of the system on several environment critical parameters constrains the system's applicability to specific environments. For instance, Ball et al. (2013) describe the procedure to configure various RatSLAM system's parameters before deploying it in a new environment. Despite following such a careful procedure, the system's performance degrades due to uncontrollable environmental factors, e.g., motion blur, noise and perceptual changes alluding from dynamic objects, as experiments show in later sections.

This chapter follows on our contribution from paper (Kazmi and Mertsching 2016a), presented at International Conference on Bio-inspired Information and Communi–cations Technologies, USA. To address the aforementioned challenges, we present a novel bio-inspired model that incrementally learns the visual representations of places. For this purpose, human-like layout of the scenes is extracted from im–ages using gist features – the biological aspect is discussed in Section 2.7.1. As this representation is not self-sufficient to model the environment, we introduce a modified GSOM and imitate the behavior of familiarity and recency neurons, see Section 2.7.3, for robust place learning and recognition. The proposed framework compactly learns the scene representations such that the close-by places are mapped to a single neuron, whereas perceptually distant locations are exclusively assigned to other neurons.

The rest of this chapter is organized as follows: Section 4.1 presents a brief modular description of the RatSLAM system. Section 4.2 introduces the proposed GSOM-based place learning front-end for RatSLAM. Regarding the procedure to extract the gist

features from images, we refer to an earlier Section 2.6.3. Section 4.3 discusses the proposed place recognition methodology and states the procedure of integrating the proposed method with the RatSLAM system as a data association module. A thorough evaluation of the proposed algorithm's performance is presented in Section 4.4, including a comparative analysis with the RatSLAM algorithm. Finally, Section 4.5 summarizes the contributions of this chapter.

4.1 RatSLAM System: A Bird's-eye View

The RatSLAM system is composed of three main components, namely *local view cells* (V), *pose cells network* (P) and the *experience map (E)*. These modules are illustrated in Figure 4.1.

4.1.1 Local View Cells

Local view cells is an array of visual templates which represent the visited places as an intensity profile. Regarding the procedure of obtaining the intensity profile from input images, we refer to (Milford and Wyeth 2008, Section VI-A). If the intensity profile of the current image matches any of the stored intensity profiles in the database, the corresponding cell is set active to inject the activity V_i in the

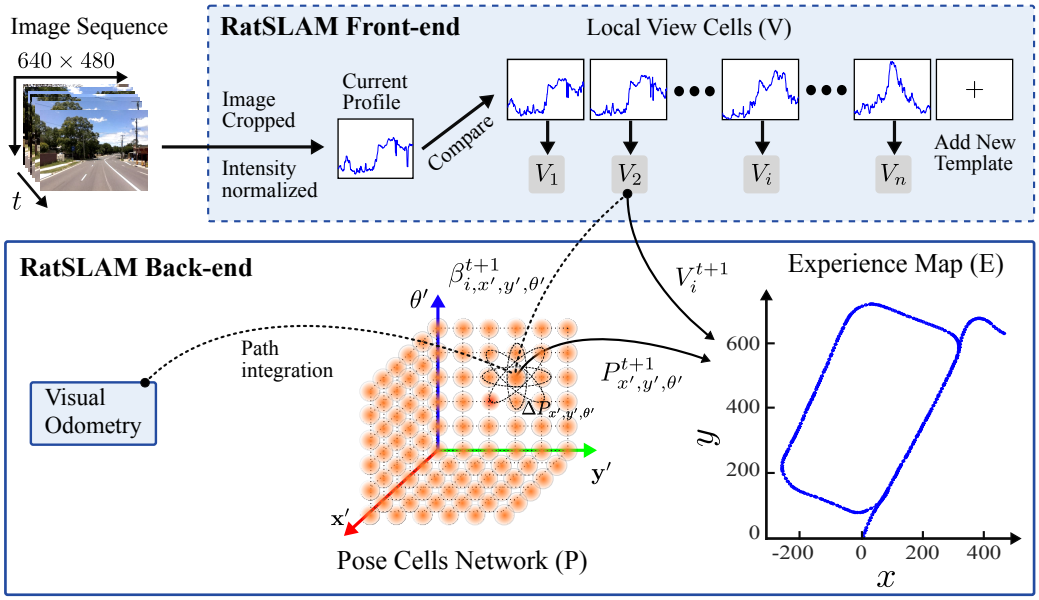


Figure 4.1: Modular description of the RatSLAM system (Milford and Wyeth 2008). Follow the anatomic details of the system in text.

pose cells associated with that scene. A formulaic description of the new connection strength $\beta^{t+1}_{i,\,x',\,y',\,\theta'}$ between the activated local view cell and the pose cell is:

$$\beta^{t+1}_{i,\,x',\,y',\,\theta'} = max(\beta^{t}_{i,\,x',\,y',\,\theta'}, \lambda V_i P_{x',\,y',\,\theta'}) \tag{4.1}$$

where the weight $\beta^{t+1}_{i,\,x',\,y',\,\theta'}$ determines the connection strength between i^{th} local view cell V_i and the pose cell $P_{x',\,y',\,\theta'}$, which is indexed by (x', y', θ') in a pose cell matrix. The parameter λ is a learning constant. A novel scene is stored as a new visual template in the local view array in the order of relative position as it is visited.

4.1.2 Pose Cells Network

A pose cell network is a three-dimensional grid of CANN, representing the weakly-metric position $(x',\ y')$ and orientation (θ') of the vehicle. The activity in pose cells is adapted via self-motion cues (here visual odometry), and re-calibrated (i.e. correction of path-integration error) with the feedback from local view cells, as follows:

$$\Delta P_{x',\,y',\,\theta'} = \sum_{i=0}^{n_{x'}-1} \sum_{j=0}^{n_{y'}-1} \sum_{k=0}^{n_{\theta'}-1} P_{i,j,k}\ \varepsilon_{a,b,c} \tag{4.2}$$

The constants $n_{x'}$, $n_{y'}$ and $n_{\theta'}$ refer to the network size along x', y', and θ' dimensions of the pose cells matrix. Whereas $\varepsilon_{a,b,c}$ is an excitatory weight mask of three-dimensional Gaussian distribution. For the definition of the mask indices a, b and c, see (Milford and Wyeth 2008, Eq. (3)). A group of highly active cells defined by this neighborhood function forms a cluster of activity. The activity injected into the network may lead to the creation of multiple packets competing to win, while the weakly active cluster of neurons gradually disappear subject to the learning rate and further feedback from vision. This process of global inhibition[1] is given by:

$$\Delta P_{x',\,y',\,\theta'} = \sum_{i=0}^{n_{x'}-1} \sum_{j=0}^{n_{y'}-1} \sum_{k=0}^{n_{\theta'}-1} P_{i,j,k}\psi_{a,b,c} - \varphi \tag{4.3}$$

where, $\psi_{a,b,c}$ is the inhibitory weight mask with negative connections and φ is the global inhibition constant. Note that Eq. 4.2 and 4.3 are the three-dimensional convolution of matrices. As this is a computationally intensive procedure, the authors achieve significant speedups by appropriately displacing the copy of activity using odometry updates, and thus sacrifice the biological fidelity.

[1] Global inhibition ensures that without visual or path-integration input the activity will eventually stabilize to one cluster.

4.1.3 Experience Map

Each experience e_i in the experience map E collectively stores the pose of the vehicle P^i and the associated local view V^i observed at that pose:

$$e_i = \{P^i, V^i, \mathbf{p}^i\} \tag{4.4}$$

Here, $\mathbf{p}^i$ is the position of the experience i^{th} experience in the experience map. For an in-depth review of the aforementioned concepts, we refer to (Milford et al. 2005; Milford and Wyeth 2008).

4.2 Proposed Model for Visual Learning

For a self-organizing maps (SOM), one has to specify the network size before the learning starts. This is unfortunately not possible when nothing is known about the environment and one has to learn the maps of visual representation from a continuous image stream. Making assumption on the size of the network may constrain the ability of the network to learn new structures in the data. Furthermore, such a network may not cope with the complexity of the environments originating from dynamic objects, appearance changes and increasing scale.

Hence, we propose a modified model of GSOM (Alahakoon et al. 2000) to incremen–tally learn the gist-based appearance of places. The network dynamics of GSOM are essentially the same as SOM except that the number of neurons in such a network grow over time to compactly learn the topological representation of the feature space. Typically, quantization error is maintained for each neuron to control the spread of the network. These attributes make GSOM superior to SOM for designing an incremental place recognition system. Unlike other machine learning algorithms (e.g. PCA), which can readily be applied to other tasks, one complication in using GSOM is that they are designed according to the task at hand. Hence, existing GSOM-based models cannot be readily applied to solve heterogeneous tasks. Al–though the general guidelines to design the GSOM model and its learning dynamics remain the same, there exist several design decisions that the data scientist has to describe according to the given task. This will become apparent to the reader in the subsequent sections.

In the following discussion, we present the proposed formulation of GSOM to incrementally learn from images. As several steps in training a GSOM are analogous to SOM, we suggest the reader to pre-read Section 2.7.2 for a better comprehension of the subsequent concepts. For the sake of clarity, we will use distinguish between

present time t and an arbitrary time instance k, where $k \in \{1, \ldots, t\}$. Say, at any time instant k, a frame $I^{(k)}$ is retrieved and its D-dimensional gist features $\mathbf{g}^{(k)} = \{g_j\}_{j=1}^{D}$ are computed, as described in Section 2.6.3. Thereafter, the input vector $\mathbf{g}^{(k)}$ is presented to the network and later discarded once the network has learned from it. Unlike in a SOM, the size of the GSOM network evolves over time. Hence, the network size at a particular time instant k will be denoted by $N^{(k)}$. We then describe the current network's state $\mathcal{S}^{(t)}$ with a $D \times N^{(t)}$-dimensional weight matrix $\mathbf{W}^{(t)}$ of neurons, a codebook comprising all the place-to-neuron mappings $\Omega^{(t)}$ and the parameters set $\Phi^{(t)}$ that govern the network dynamics:

$$\mathcal{S}^{(t)} = \left(\mathbf{W}^{(t)}, \Omega^{(t)}, \Phi^{(t)}\right) \tag{4.5}$$

A neuron i at a particular time instance k, where $1 \leq i \leq N^{(k)}$, is described by: (a) a D-dimensional weight vector $\mathbf{w}_i^{(k)}$, which is the i^{th} column vector of the weight matrix, (b) a bounded but decaying learning rate $0 < \alpha_i^{(k)} \leq \alpha_0$, where $a_0 < 1$ is an initial learning rate, and (c) a set of places learned by the neuron $\omega_i^{(k)} \subset \Omega^{(k)}$.

Network Initialization

The two important steps for the network initialization include: a) the initial network size and b) the procedure of initializing weights of start-up neurons in the network. We initialize the network with a single neuron and populated its weight vector with the gist features of the first frame. Finally, the weight vector is normalized to the unit length. This choice of weights initialization justifies, as images arrive in a stream and no knowledge of the environment is available beforehand.

Determine the Winner Neuron

At the present time instant t, a feature descriptor is fed to the network and its distance to each neuron is calculated. A neuron c is deemed a best matching neuron (or a winner), if its spatial characteristics closely match the perceptual description of the current place. There exist different choices for selecting a distance measure, as already discussed in Section 2.7.2, which typically depends on the task at hand. For the recognition tasks, the sum of squared difference has been frequently adopted (Aly et al. 2009; Chen et al. 2015; Oliva and Torralba 2001):

$$d_s(t, i) = \sum_{j=1}^{D} \left(g_j^{(t)} - w_{ji}^{(t)}\right)^2 \tag{4.6}$$

where $d_s(t, i)$ is the component-wise sum of squared differences between the i^{th} neuron's weight vector and the current gist descriptor. Conventionally, the winner

neuron is determined by minimizing Eq. (4.6) for all neurons $i \in N^{(t)}$. However, due to high visual ambiguity in natural environments, one may find perceptually similar visual descriptors for two locations that are physically distinct. This could result in false positive detections. To mitigate such a odds of perceptual aliasing, we also consider the distances of the neighbors of the neuron i from the descriptor and minimize the following objective function for selecting the winner c:

$$c = \arg\min_{i} \left(d_s(t,i) + \frac{1}{N_i^{(t)}} \sum_{p \in N_i^{(t)}} d_s(t,p) \right) \tag{4.7}$$

where $N_i^{(t)}$ is the neighborhood of the neuron i. In the above equation, the summation term incurs the cost on assigning a wrong neuron to the feature descriptor of current location. The effect of the above expression is that all the places for which c is the winner neuron are exclusively mapped to that neuron, denoted as $\omega_c^{(t)} \subset \Omega^{(t)}$; here $\omega_c^{(t)}$ is a set of those gist descriptors that are learned by the winner up to present time. One should, however, note that finding the optimal winner neuron cannot always be guaranteed in natural scenes, as they share ample perceptual similarity. Like RatSLAM, the computational complexity of our algorithm to find the best neuron is $\mathcal{O}(DN^{(t)})$, but our approach facilitates RatSLAM to represent the same environment with only few experience cells. This leads to the reduced size of the search space and thus efficient retrieval of the winner neuron.

Adapting Network Weights

Once the winner neuron c is known, the next step is the adaptation of network weights. Typically, one updates the weight of the winner and its topological neighbors. The places in the space of gist features are spread contiguously along the perceptual axes (Oliva and Torralba 2001). This suggests that physically nearby places will appear next to each other in space of gist features, as they share similar perceptual attributes. On that account, we only adapt the weights of the winner:

$$\mathbf{w}_i^{(t+1)} = \begin{cases} \mathbf{w}_i^{(t)} + \alpha_i^{(t)} \left(\mathbf{g}^{(t)} - \mathbf{w}_i^{(t)} \right) & i = c \\ \mathbf{w}_i^{(t)} & \text{otherwise} \end{cases} \tag{4.8}$$

The above equation is also referred to as a hard competitive learning, aka winner-–takes-all mechanism, as only the winner adapts its weights for the given input vector. The learning rate $\alpha_i^{(t)}$ of a neuron is often set to decay over time, see for instance Section 2.7.2. We rather decrease the learning rate as a function of neighborhood size of the winner c:

$$\alpha_c^{t+1} = \alpha_0 \exp\left(-\frac{N_c^{(t)}}{\sigma} \right) \tag{4.9}$$

where the initial learning rate α_0 and the maximum allowed neighborhood size of the winner σ are empirically set to the values 0.1 and 4, respectively. The learning rate decays as more neurons grow in the neighborhood of the neuron. Finally, the adapted weight vector is constrained to the unit length, i.e., $\left\|\mathbf{w}_i^{(t)}\right\| = 1$.

Growing a New Neuron

The process of growing a new neuron needs the consideration to the following aspects: (a) when to grow a new neuron?, (b) the insertion location of the new neuron, and (c) initialization of the new neuron's weight vector. We decide on growing a new neuron if the distance of the current feature descriptor from any neuron i, given by Eq. (4.6), exceeds a preset threshold value $\delta = 0.07$. During the exploration, a vehicle tends to visit the physically, as well as perceptually, nearby locations in an ordered sequence. Thus, a new neuron is inserted next to the closest neuron. In case, two neurons are equally close to the current gist descriptor, a new neuron is inserted exactly between those neurons. Finally, the weight vectors of the new neuron are initialized with the gist features of the current descriptor.

4.3 Leveraging Proposed Model for Place Recognition

It is apparent from the Section 4.1 that the correct place recognition, i.e., loop detection, is vital to robust pose estimation. A wrong pose association may otherwise lead to a degenerated map. The existing place recognition front-end of the RatSLAM system does not perform learning. It rather creates visual templates based on a pre-selected threshold value. Moreover, representing images with the intensity profile is not robust, as an image's brightness level is significantly affected by the presence of dynamic objects or illumination changes. Hence, we present a novel place recognition front-end for the RatSLAM system, see Figure 4.2, which exploits gist features to capture the spatial layout of the scenes. The extracted features are fed to a growing network of self-organizing neurons, which compete with one another and adapt their positions in the feature space to coarsely learn the topological structure of the data. We do not acquire any prior knowledge of the environment, and thus achieve on-the-fly scene learning and recognition.

Given the gist descriptor $\mathbf{g}^{(t)}$ at a present time t, a neuron is said to be familiar with the current location, if it has the least perceptual distance to this place amongst the existing neurons. In the present setup, we set an upper bound δ on the minimum distance. Otherwise, the recognition system may pickup on the spurious inputs

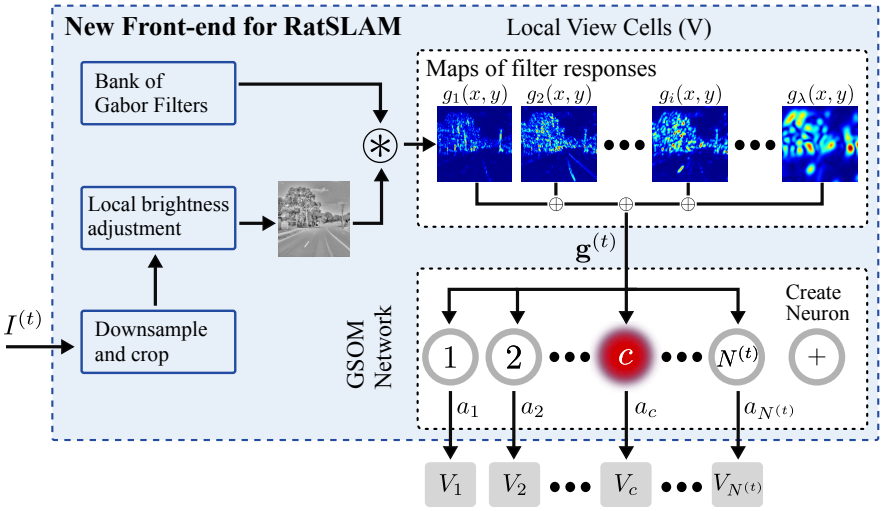


Figure 4.2: GSOM-based place recognition front-end for RatSLAM. Given an input image, its gist descriptor is computed and fed to the network. A neuron that is familiar to this input will show high activity, e.g., the one indicated in red. Whereas a novel input will result in creation of a new neuron to learn from the input pattern. Note that gist features are computed in frequency domain. This figure displays the filter responses in spatial domain only for visualization purposes.

and flag them familiar. The neuron that shows familiarity to the input gets active $a_i = 1$, whereas other neurons remain inactive. The activity level of the neuron V_i is obtained from the frequency variable, which is incremented every time a neuron is activated. Upon experiencing a novel scene, a new neuron is created, as in this case the distance to the closest neuron will exceed the δ. Finally, this information travels down to the back-end, i.e., pose cells network and the experience map, to carryout the localization and mapping steps.

4.4 Evaluation and Results

This section first describes the experimental configurations. Thereon, various test cases are considered for evaluating the performance of the proposed model and RatSLAM's baseline method for place recognition.

4.4.1 Experimental Setup

For experimental evaluation of the proposed approach, the Loop test video sequence (RatSLAM 2009) is used as a benchmark dataset. The video is recorded at a frame rate of 25 fps, which yields ca. 2517 frames having resolution of 640×480 pixels. As the ground truth data is not available, we utilized the timestamps to tag the frames as visited or unvisited. We noticed that the vehicle re-visits the previously explored route at 1:09 (mm:ss) and it continues driving along that route for about 30 seconds.

Hence, the frames that fall within the indicated timestamps are tagged as visited ground truth locations, while the remaining are considered unvisited. To further localize the accuracy of loop locations, we set the margin of one second around the individual frames. The algorithm is implemented and tested in MATLAB, which runs on a 3.3 GHz Intel Core i5 processor with 8 GB physical memory.

4.4.2 Evolution of View Cells: Proposed vs. Baseline Model

To compare our place recognition module with RatSLAM's baseline model, we observed the evolution of the view cells and particularly looked out for the false positive and negative predictions of the algorithm. Amongst the two kinds of errors, false positive predictions have severe consequences compared to false negatives. A false positive prediction of the algorithm associates an unvisited location to a previously visited location. This can collapse a navigation system. On the other hand, an algorithm's prediction is false negative when it misses the detection of a visited location (i.e. loop-closure).

Figure 4.3 shows the activity in view cells as a function of frames. The very first difference that can be noticed between the two approaches is number of view cells created to represent 2517 frames. The GSOM-based place recognition model grows only 297 neurons to learn the environment, whereas RatSLAM's baseline front-end represented the same sequence with 541 cells. This indicates that the proposed

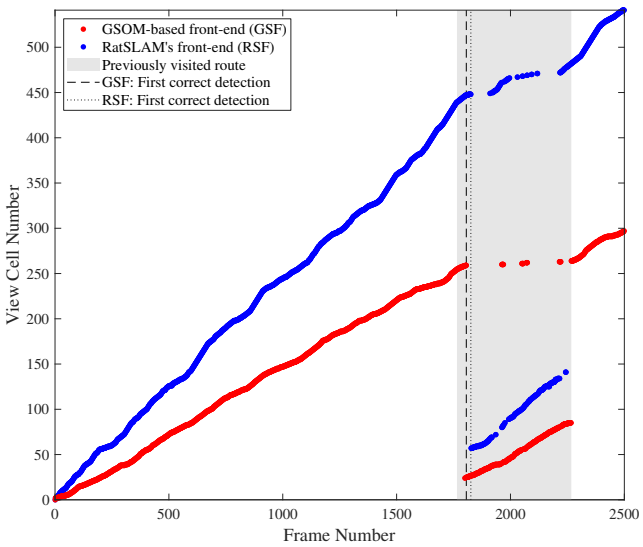


Figure 4.3: Evolution of view cells in proposed and baseline methods. The gray column refers to the loop closure region, representing activations (or recalls) of cells in network (see text for details). The loop detection latency of GSOM-based front-end is ∼1.6 seconds (i.e. ca. 40 frames) in comparison to ∼2.4 seconds latency time for the baseline method, indicated by the vertical bars for the respective methods (Kazmi and Mertsching 2016a).

approach requires less physical memory compared to existing RatSLAM's front-end, and thus it can potentially be employed for loop detection on longer routes.

Inside the loop closing region, i.e., the gray shaded area along the vertical axis in the figure, follow the sequence of points that fall below the progressing sequence. These points indicate the selection of previously created view cells. Amongst the two methods, it can be noticed that GSOM-based front-end showed high true positive rate compared to the baseline place recognition, as more points fall back along the vertical axis as a function of new frame. Moreover, the RatSLAM's baseline created more cells in the revisited region. This suggests that the intensity profile is not robust to momentary changes in layout of the scene that originate from the moving objects or shadows of the trees along the routes.

The dashed vertical line in the loop closing region refers to the frame when first correct recall (i.e. true positive) is generated in GSOM-based front-end, while the dotted line indicates the first true positive recall for the baseline algorithm. Note the latency in true positive detection of the two systems. Indeed, the loop detection latency delays the correction of drifts in the map. A small gray region before the dashed and dotted lines of the respective algorithms is the loop closing area that remained undetected by both the algorithms due to high viewpoint changes, while the span of the undetected region (false negatives) is larger for the baseline approach, compared to our method.

4.4.3 Robustness to Sensory Perturbations

In vision-based systems, the quality of images recorded with a consumer camera remains susceptible to many uncontrollable external factors, such as particulates and motion blur due to in plane camera rotations or low frame rate. These perturbations may lead to false predictions (i.e. false positive or negative). Hence, we evaluate the robustness of the proposed and baseline systems in presence of noise and motion blur. To mimic the former case, white Gaussian noise is added to every input image before feature extraction. Fixing $\mu = 0$, the variance σ^2 is seeded randomly such that $0.01 \leq \sigma^2 < 0.1$. For the latter case, the motion blur is applied to images in the direction opposite to the camera rotation, where visual odometry of the RatSLAM system is utilized to obtain the camera rotation between two consecutive movements. The example images for the additive white Gaussian noise and motion blur are depicted in Figure 4.4 alongside the original frame.

In the presence of white noise, GSOM-based front-end creates 345 view cells, whereas RatSLAM's existing front-end allocates 545 cells to represent the same image stream, see Figure 4.5. This means that in relation to no noise case, the

(a) Original Image **(b)** Gaussian noise **(c)** Motion blur

Figure 4.4: Image shown in (a) is frame# 1947 of the St. Lucia Loop test sequence. The other two images, i.e., (b) and (c), are obtained by applying zero mean Gaussian noise ($\sigma = 0.1$) and in plane rotation of 1°, respectively, to the original image (Kazmi and Mertsching 2016a).

growth of the view cells for the proposed approach has increased by a factor of $\sim$1.2, which by contrast has not changed for the baseline method. We define the growth factor as a ratio of the view cells created in noise and no noise cases. This does not by any mean imply that baseline is robust in presence of noise, rather we should look for the loop detection latency and the false negative rates.

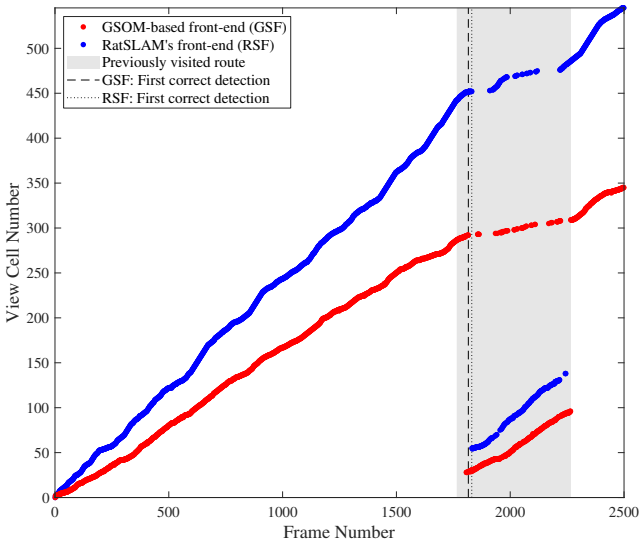


Figure 4.5: Evolution of view cells in the presence of white noise. Despite noisy input, the proposed method offers fast loop detection compared to baseline, e.g., note the difference in position of the dashed and dotted vertical lines on the horizontal axis (Kazmi and Mertsching 2016a).

The reader should furthermore note that the baseline method computes the nor–malized sum of the intensities along the column of the cropped region of interest within an image, which in effect compensates the net disturbances due to noise along the vertical image axis. By contrast, gist features capture both the low and high frequency content in images. As white Gaussian noise substantially degenerates the high-frequency information (i.e. edge distribution), the presence of white Gaussian noise tend to diffuse the filter responses. This explains the rational for creating more view cells in case of gist features in presence of white Gaussian noise. To reduce the

impact of noise, images can be preprocessed with a lowpass filter before extracting the gist features. In the figure, it can also be noticed that compared to no noise case, the loop detection latency has slightly increased for both the algorithms, which indicates more false negatives (i.e. less true positives) and thus relatively lower recall rate.

While additive white Gaussian noise increases the false negative rate, motion blur elevates the false positive rate. Due to the fact that motion blurring smooths images, high-frequency information fades out. This may result in perceptual aliasing because the high-frequency content smudges with the low frequencies in image. Consequently, it is more likely to pickup on a false match. For instance, the visual description of two physically distant locations may look similar in smoothed images. This effect can be observed in Figure 4.6, see e.g., the increase in number of false positives. In the figure, false positive recalls are the points outside the loop closing region that fall below the progressing diagonal. It can be noticed in the figure that the number of false positive predictions are more in case of RatSLAM, whereas the motion blur has only a mild impact on GSOM-based front-end.

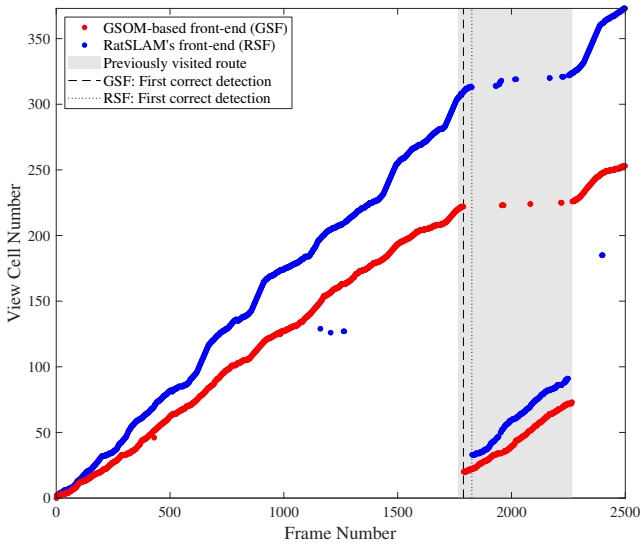


Figure 4.6: Applying motion blur increases the false positive activity in view cells, which is more noticeable for the RatSLAM's existing front-end, while proposed algorithm relatively remains robust to perceptual aliasing (Kazmi and Mertsching 2016a).

Regarding the motion blur scenario, an interesting trend in algorithms' behavior is the decline in the number of view cells created to map the environment. For example, our approach has grown 253 neurons in comparison to 372 view cells of RatSLAM's baseline method. Such a decline in the growth factor is an obvious response of the algorithms, as motion blur suppress the high-frequency components in images. Consequently, little information is available to discriminate between the the scenes. Besides this, relative to the first correct loop detection of GSOM-based algorithm, the RatSLAM's first correct detection has delayed by $\sim$1.5 seconds. For

example, follow the distance between the dashed and dotted vertical lines. Indeed, this gap between the two algorithms was previously less than a second.

4.4.4 Precision–Recall Scores

For quantitative evaluation of the GSOM-based front-end (GSF) and RatSLAM's front-end (RSF), we utilize the precision-recall scores and F_1-measure. A detailed explanation of the evaluation metrics is presented in Section 2.5.4. The precision–recall scores of the algorithms are computed using Eq. (2.8) and Eq. (2.9), whereas for F_1-score is obtained from the precision and recall scores according to Eq. (2.10). Computing above scores depends on the definition of TP, FP and FN cases. An algorithm's prediction is regarded as TP, if it correctly recognizes a visited location. The prediction of the algorithm is referred to as FN, if it classifies the visited location as an unvisited and it is FP when algorithm predicts an unvisited location as visited. Figure 4.7 plots the precision–recall and F_1-scores of the baseline (RSF) and proposed (GSF) algorithms in no noise scenario, including their performances in the presence of motion blur, referred to as RSF+Blur and GSF+Blur, and white noise, abbreviated as RSF+WN and GSF+WN, respectively.

In no noise case, GSF (ours) and RSF show comparable precision scores of 100% and 99%, respectively. However, a huge difference is notable in the recall rate of the two systems, i.e., GSF: 87.88% and RSF: 63.4%. Adding white noise to the images

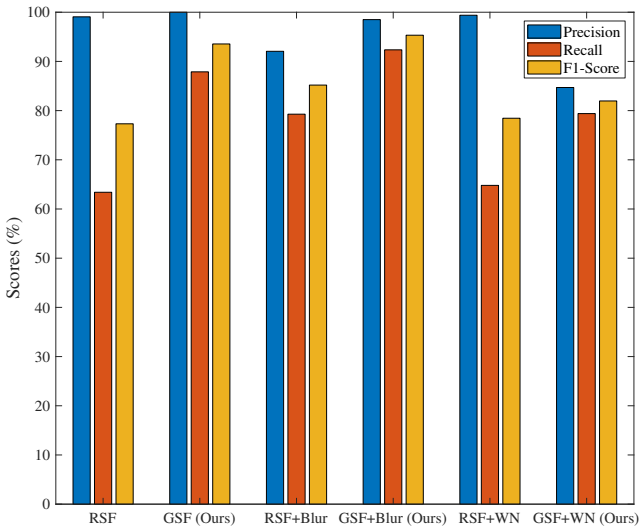


Figure 4.7: Precision–recall and F_1-scores in various test scenarios: Our approach (GSF) outperforms the baseline (RSF) in nearly all experiments. In case of white noise, RSF+WN achieves higher precision 99.39% but its recall is merely 64.8% compared to a 79.4% recall rate achieved by our GSF+WN. The use of F_1-measure demonstrates the balance between false positive and negative predictions, in which case our algorithm outperforms (Kazmi and Mertsching 2016a).

does not drop the precision of RSF+WN (99.39%), whereas the precision score of GSF+WN (ours) gets relatively low, i.e., 84.7%. The reason for such a drop in precision is already discussed in Section 4.4.3. One should however note that in comparison to 64.8% recall rate of the baseline method, our algorithm achieves a recall of 79.4% in this experiment. Essentially, a low precision score is the consequence of high false positive rate, whereas low recall originates from high false negative rate. Hence, algorithms are usually ranked using F_1-measure to make a balance between precision and recall rates, as an algorithm with high precision and very low recall or vice versa does not deliver robust solution to place recognition. Hence, we computed the F_1-score of the algorithms using respective precision–recall scores. Regarding F_1-measure, our algorithm outperforms the baseline with scores: GSF+WN (ours): 81.96% and RSF+WN: 78.45%. In case of a motion blur, our approach has outperformed the baseline approach.

Finally, we integrate the proposed front-end with RatSLAM system (GSF+RatSLAM) and performed the topological mapping of the environment, as shown in Figure 4.8. At a first glance, it is apparent that both the algorithms were able to detect loop locations and retained the topology of the places traversed by vehicle. However, in Sections 4.4.2 and 4.4.3, we showed that our method comparatively offers a faster loop detection. Moreover, proposed algorithm learns a compact representation of the environment, as only few neurons are created to represent the underlying test environment.

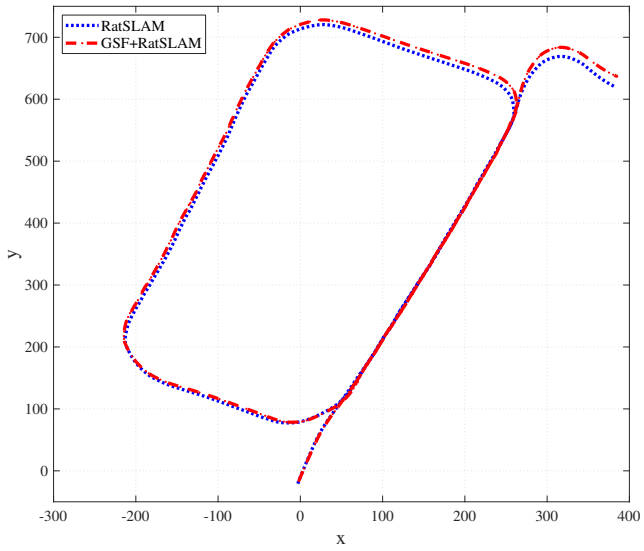


Figure 4.8: Topological map of the route traversed by the vehicle. Both RatSLAM and GSF+RatSLAM corretly detect loop locations and preserve topology of visited places. However, GSF+RatSLAM demonstrates fast loop detection (Kazmi and Mertsching 2016a).

4.5 Summary

The contributions presented in this chapter originate from (Kazmi and Mertsching 2016a). This research work takes the inspiration from biology to design a technical system that can learn and recognize unknown environments on-the-fly. For this purpose, human-like representation of the scene, referred to as gist, is extracted from images. The extracted scene representations are learned incrementally using a modified growing self-organizing neural network, which mimics the competitive behavior of cells found in cortical regions of the brain. Finally, the proposed system is tested on the benchmark dataset by simulating white noise and motion blur in images. The results of the experiments demonstrate that proposed approach in general achieves high precision–recall performance compared to baseline place recognition system. This also highlights the strength of our approach to recognize places in previously unobserved scenarios. Moreover, the algorithm's loop detection latency remained less than approx. 1.6 seconds compared to baseline method in all experiments. On the average, our algorithm creates 20% less view cells than existing RatSLAM's front-end. Hence, the proposed algorithm enforces considerably low memory requirements than baseline.

5 Growing Belief Network for Real-time Place Recognition

In previous chapter, we proposed a bio-inspired model for on-the-fly scene learning. However, there are several aspects of the algorithm that need further consideration. For instance, previous GSOM model does not account for the progression of error in the network in relation to the fed input. Hence, the quality of the learned feature maps cannot be assessed quantitatively. Secondly, creating neurons by thresholding the input-to-neuron distances restrains the network from adapting itself as per density of the data in the feature space. Moreover, hard-competitive learning for adjusting the network weights, aka winner-takes-all mechanism, has certain drawbacks (Fritzke 1997), such as creation of neurons that exist in the network for an indefinite time. Such neurons are never used again, as they do not become winner for any input. They do not contribute to error minimization and are wasteful allocation of the resources. An indirect way to avoid this issue is to initialize the weights of new neurons from the feature space, as we followed in previous approach, but this does not solve the problem due to dynamic nature of environments. Indeed, in error minimization the goal is to undersample the high density regions in the feature space and assign more samples to the low density regions. Besides this, the cost function to determine the winner in Eq. (4.7) is computationally intensive for place recognition in large-scale scenarios. Finally, the NNS criteria to recognize familiar and novel places is so naive that it can fall into the trap of perceptual aliasing, resulting from the visual ambiguity in environments.

The contributions of this chapter stem from the paper presented at IEEE International Conference on Intelligent Robots and Systems (Kazmi and Mertsching 2016b). In this chapter, we introduce standalone algorithms for real-time place learning and recognition. Regarding visual learning, we propose novel dynamics for the existing GSOM model (Kazmi and Mertsching 2016a) and fuse this network with the proposed Bayesian network for place recognition. Altogether, we call this system a *growing belief network (GBN)*. While subtle difference in the learning dynamics of the network will become apparent in subsequent sections, here we highlight the most significant differences between our previous and new approaches: (a) Instead of hard-competitive learning, we follow a soft-competitive strategy to adapt the weights of neurons in the network. As a result, the winner neuron and its topological neighbors in the feature space adjust their weights. (b) The the network's growth is

controlled through the error in the network. This facilitates the network to expand itself dynamically according to the scale and complexity of the environment. For instance, more neurons are assigned to the unexplored regions of the feature space. (c) Inactive neurons are periodically pruned from the network. These neurons otherwise consume system's resources and even increase the winner search time. Finally, given that the neurons have coarsely learned the distribution of the feature space, (d) we maximize a posteriori over the network. This assigns certainty to our belief in classifying a query place as familiar or novel.

Subsequent discussion in this chapter continues as follows: Section 5.1 presents the conceptual formulation of the proposed system (GBN). In Section 5.2, we introduce novel learning dynamics for GSOM. Section 5.3 describes the proposed belief-based network for place recognition. A comprehensive evaluation of GBN on three challenging datasets is demonstrated in Section 5.4, which also includes the comparison of GBN with popular place recognition methods. Lastly, Section 5.5 summarizes the research problems solved in this chapter.

5.1 Introduction

To model the place recognition challenge, we take inspiration from Bayesian statistics and fuse it with the growing SOM. Altogether the system is named as growing belief network (GBN). A belief network (or Bayesian network) is a directed acyclic graph of stochastic variables. Some of these variables are observed, e.g., the data points in the feature space, while others remain unobserved to the user, referred to as hidden variables, as illustrated in Figure 5.1. The unobserved variables can be stacked into multiple layers forming a tree structure. For example, in the figure two layers case is

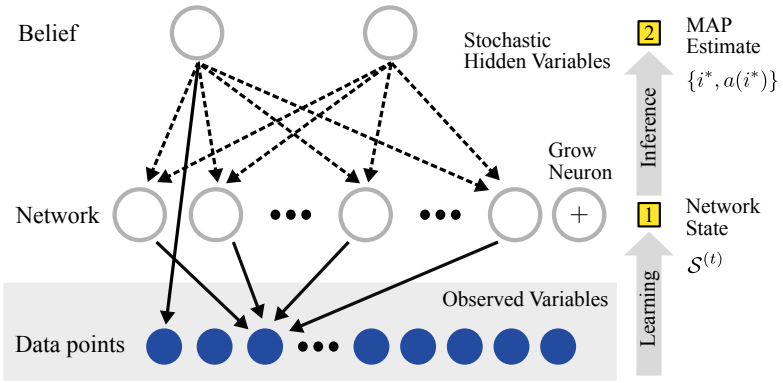


Figure 5.1: Conceptual layout of an evolving Belief network. Gray nodes in the figure are unobserved variables. The goal is to adjust the neurons' weights (or grow more neurons) in the network to maximize the likelihood of generating the observed data, while at the same time inferring the network's belief regarding query data point. Follow the details in text and Section 5.3.

demonstrated. Now the goal is to solve two problems simultaneously: (1) *Learning problem* to adjust the unobserved variables such that the likelihood of generating the observed data from the network increases. In our case, this data representation is learned through a GSOM. Hence, the neurons of the GSOM are those hidden variables in the middle layer that interact with each other to learn the representation of the input space. At the same time, (2) the *inference problem* must be addressed to determine the state of the unobserved variables. In the present case, it is analogous to inferring the belief of a neuron in response to a query image. Indeed, each neuron carries a belief in representing the fed input, which is the posterior distribution over the network. Maximizing this posterior estimate helps us to make a prediction, if the query image comes from the familiar or novel places. Note that the network evolves dynamically to learn the representation of the feature space. While evolving, the network is also expanding in the feature space. As we apply the Bayesian statistics to the growing network, we call it growing belief network for the reason that each neuron has a belief about the current state of the environment. The probabilistic model of this formulation is described in Section 5.3.

We now present the algorithmic workflow of GBN, which is depicted in Figure 5.2. The system learns from an image stream in a learning epoch. A learning epoch in our case is the temporal window of size ρ in which the network weights are adapted in relation to presented input. Thereafter, network pruning is performed. Hence, grab–bing frames in chunks and then applying network pruning is essentially equivalent to retrieving one frame at a time from the image stream and performing pruning after every ρ frames. During the learning phase, an image is grabbed from the sequence and its gist features $\mathbf{g}^{(t)}$ are obtained according to Section 2.6.3. Thereon, the ex–tracted gist descriptor is fed to the visual learning and place recognition modules. The GSOM network adjusts its weights based on error due to gist descriptor of the

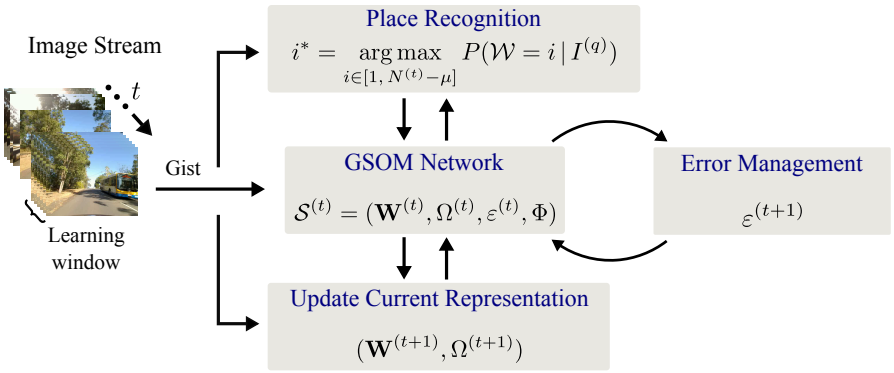


Figure 5.2: Algorithmic workflow of growing belief network (GBN). At a present time t, GBN retrievers a frame, computes its gist features and simultaneously performs scene learning and recognition. After processing the images in a learning window, network pruning is performed to prevent the unnecessary resource allocation (Kazmi and Mertsching 2016b).

current location. Whereas, the place recognition algorithm utilizes the incrementally trained network and estimates maximum a posteriori (MAP) over the network to distinguish the present image as novel or familiar.

5.2 Novel Network Dynamics for Visual Learning

At a present time t, the network's state is described by network weights $\mathbf{W}^{(t)}$ and the codebook of all the mappings to the neurons $\Omega^{(t)}$. Note that the variables and the notations used in this section are consistent with the Section 4.2. In the present setup, the network additionally accounts for error in each neuron, which is represented by an $N^{(t)}$-dimensional vector $\varepsilon^{(t)}$. Furthermore, we now fixed the learning rate compared to previous model – motivation for such a choice is discussed in Section 5.2. As a result, the parameters set Φ is no more indexed by time. The network's state is henceforth described as follows:

$$\mathcal{S}^{(t)} = \left(\mathbf{W}^{(t)},\, \Omega^{(t)},\, \varepsilon^{(t)},\, \Phi\right) \tag{5.1}$$

The contents of a neuron at a particular time instance k include: (i) a weight vector $\mathbf{w}_i^{(k)}$, (ii) a constant learning rate α, (iii) a set of places mapped to the neuron $\omega_i^{(k)}$, (iv) learning frequency[1] $f_i^{(k)}$ and (v) the quantization error $\varepsilon_i^{(k)}$. We now describe the novel network dynamics for GSOM.

Network Initialization

Two important steps for network initialization are: 1) the initial network size and b) the procedure of initializing weights of start-up cells in the network. We start with two neurons in the network and initialize their weight vectors using the gist features of starting frames:

$$\mathbf{w}_i^{(t)} = \gamma\, \mathbf{g}^{(t)} \tag{5.2}$$

where $\gamma \in (0, 1]$ is the neuron creation weight. It determines how close the neuron will be instantiated relative to the current descriptor. For $\gamma = 1$, the weights of the new neuron will be the replica of gist features of the current location. However, this may lead to over-fitting the input space. By contrast, small γ values induce instability, as such a choice will place the neurons too far from the actual input. The rest of the input proportion is accumulated in the corresponding error of the neuron:

$$\varepsilon_i^{(t)} = (1 - \gamma) \sum_{j=1}^{D} \left(g_j^{(t)} - w_{ji}^{(t)}\right)^2 \tag{5.3}$$

[1]Learning frequency is the number of times a neuron is deemed winner in the network.

Determine the Winner Neuron

We use the sum of squared differences metric, given by Eq. (4.6), to compute the image-to-neuron distances and find the winner c as follows:

$$c = \arg\min_i \left\| \mathbf{g}^{(t)} - \mathbf{w}_i^{(t)} \right\|^2 \tag{5.4}$$

Hence, the set of all mappings to the winner $\omega_c^{(t)}$ are given by:

$$\omega_c^{(t)} = \left\{ \mathbf{g}^{(k)} : \left\| \mathbf{g}^{(k)} - \mathbf{w}_c^{(t)} \right\|^2 \leq \left\| \mathbf{g}^{(k)} - \mathbf{w}_i^{(t)} \right\|^2 \right\} \tag{5.5}$$

The above expression states that the places for which c is the winner neuron are exclusively mapped to it. This concept is illustrated in Figure 5.3. The selection of a neuron as a winner c increases its learning frequency $f_c^{(t+1)}$ by one.

Adapting Network Weights

Preserving the topology of the feature space substantially rely on the learning procedure. To this end, we follow a soft-competitive learning procedure, aka winner-takes-most mechanism. Hence, the weights vectors of the winner neuron as well as its topological neighbors are updated:

$$\mathbf{w}_i^{(t+1)} = \mathbf{w}_i^{(t)} + \alpha\, \varphi\left(\mathbf{w}_i^{(t)}, \mathbf{w}_c^{(t)}\right)\left(\mathbf{g}^{(t)} - \mathbf{w}_i^{(t)}\right) \tag{5.6}$$

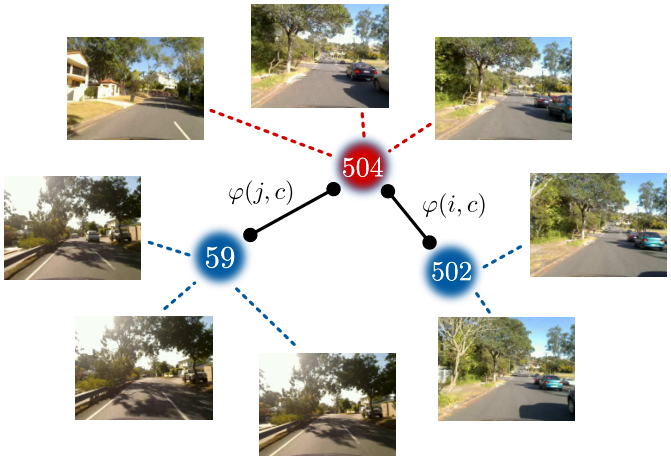


Figure 5.3: Place-to-neuron mappings in GSOM (Kazmi and Mertsching 2016b). The shown neurons are the part of the network which has learned the visual representation of the St. Lucia suburb. Given the winner c: 504, the neuron j: 59 receives zero excitation from the winner. On the contrary, neuron i: 502 shares an excitatory connection with the winner, i.e., $\varphi(i,c) \approx 0.18$, and so it is a topological neighbor of c.

where parameter α is the constant learning rate and the kernel $\varphi(.,.)$ is the Gaussian radial basis function (RBF). Setting a small fixed learning rate keeps the network adaptive and allows gradual recovery from the local minima (Fritzke 1997). One might have noticed that the above equation is a stochastic learning process, as the network's weights are adjusted at a single example. Thus, large learning rate must be avoided for it will drag the network away from the global optimum. Consequently, the network's activity will either take longer to stabilize or hardly show convergence to a stable pattern of activity.

The Gaussian RBF in Eq. (5.6) is centered around the winner neuron c. Generally, two arbitrary neurons i and j are topological neighbors, if the following satisfies, i.e., $\varphi(\mathbf{w}_i, \mathbf{w}_j) > 0$. Otherwise, they have zero excitation strength between them. We define the Gaussian RBF as follows:

$$\varphi(\mathbf{w}_i, \mathbf{w}_j) = \exp\left(-\beta \left\|\mathbf{w}_i - \mathbf{w}_j\right\|^2\right) \tag{5.7}$$

The parameter $\beta \in (0, D]$ is the kernel's bandwidth around the winner. It determines the spatial interaction among neurons. Figure 5.4 plots the response of the Gaussian RBF for two pairs of neurons at different β values. Consider c: 504 as a winner neuron. Based on RBF response, it is clear that i: 502 is a topological neighbor of the winner, whereas neuron j: 59 is not because of zero-excitation between them. Reducing the bandwidth, i.e., $0 < \beta \leq 32$, makes j and c the topological neighbors. Nevertheless, the excitation strength between them is much weaker than the connection between c and i. From the plot, it is apparent that large β values reduce the connectivity in network. Consequently, the neurons in the network remain isolated and so more neurons are needed to learn the representation of the environment. On the contrary, too low values of β induce interaction between all the neurons, causing the network to oscillate. That is to say, the neurons keep dragging themselves between the inputs subject to their connection strength to the

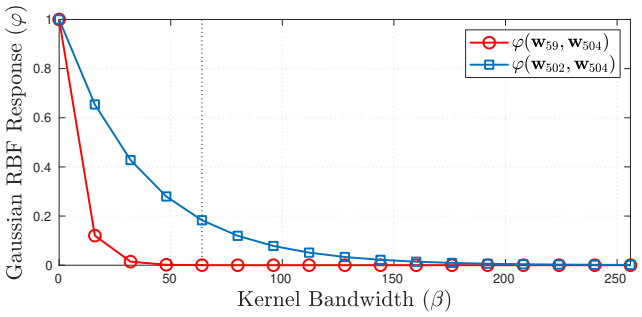


Figure 5.4: Response of Gaussian RBF for different values of kernel bandwidth. The bandwidth is varied in the range $[0, \frac{D}{2}]$ with a step size of 16. The weight vectors $\mathbf{w}_i$ belong to the network that has learned the representation of St. Lucia suburb. Assuming c: 504 as a winner, it can be noticed that j: 59 is not the topological neighbor of c at $\beta = 64$ because of zero-excitation between them.

winner. Note that at $\beta = 0$ both i and j are treated equally. It is the case when kernel becomes ineffective. After following the weight adjustment procedure, we normalize the weight vectors to a unit length, i.e., $\left\|\mathbf{w}_i^{(t+1)}\right\| = 1$.

Controlling the Network Growth

Instead of thresholding the input-to-neuron distance to control the network's spread, we regulate the error in the network. When quantization error in the winner neuron due to current input exceeds the minimum error tolerance, i.e., $\varepsilon_c^{(t+1)} > \tau$, a new neuron is instantiated in the network. Where $\varepsilon_c^{(t+1)}$ is the sum of squared error (SSE) in the winner up to present time. It is given as:

$$\varepsilon_c^{(t+1)} = \varepsilon_c^{(t)} + \sum_{j=1}^{D} \left(g_j^{(t)} - w_{jc}^{(t)}\right)^2 \tag{5.8}$$

To avoid over-fitting the feature space, the weights of the newly instantiated neuron are populated according to Eq. (5.2) and the corresponding quantization error is initialized using Eq. (5.3). Thereon, we smoothed out the network error by a small amount $\varepsilon^{(t+1)} = \eta\, \varepsilon^{(t)}$, where $0 < \eta < 1$, which assists the network dynamics against populating dense regions of the feature space with too many neurons.

Network Pruning

After each learning epoch, where the size of the learning epoch is defined by ρ, the spurious neurons are removed from the network. Indeed, these are the neurons that are created from the bogus input, e.g., due to dynamic objects in the scene. Despite the fact these neurons are never used again, they still occupy the the system's resources and increase the time to search a winner neuron. To pick out the spurious neurons, we utilize the neurons' learning frequency and prune the ones with zero learning frequency. Finally, re-indexing is performed for a subset of recently created neurons whose indexes are affected by the deletion process and network's state is updated accordingly.

5.3 Probabilistic Modeling of the Belief Network

This section draws upon the conceptual formulation of the growing belief network that we set earlier in this chapter, see Section 5.1. Let $\mathcal{W}$ be a discrete random variable that instantiates to the neuron's index in the network. Given the current observation of the world $I^{(q)}$ and the representation learned by the network $\mathcal{S}^{(t)}$ up to present time, the goal is to make an inference if the current observation

comes from the familiar places or it is a novel location. To this end, we maximize a posteriori (MAP) over the network:

$$i^* = \underset{i \in [1,\, N^{(t)} - \mu]}{\arg\max} \; P(\mathcal{W} = i \,|\, I^{(q)}) \tag{5.9}$$

where i^* refers to the index of the maximizer neuron, i.e., the neuron that maximizes a posteriori estimate for the query frame. Note that the posterior distribution is computed for all the neurons except μ recently instantiated neurons. Indeed, the learning frequency of the recently created neurons could be zero or they are more likely to resemble the current observations. Hence, these neurons neurons should contribute to the inference of place recognition. In Eq. (5.9), the right-hand side can be expressed with the Bayes rule, as follows:

$$P(\mathcal{W} = i \,|\, I^{(q)}) = \frac{P(I^{(q)} \,|\, \mathcal{W} = i) P(\mathcal{W} = i)}{P(I^{(q)})} \tag{5.10}$$

The denominator term is the marginal probability of the query frame with respect to the current network state, i.e., $P(I^{(q)}) = \sum_{i'} P(I^{(q)} \,|\, \mathcal{W} = i') P(\mathcal{W} = i')$. Hence, it can be treated as a normalization constant. The class conditional probability $P(I^{(q)} \,|\, \mathcal{W} = i)$ for a neuron i is the likelihood that this neuron generates the best visual description of the query frame. We model this term as follows:

$$P(I^{(q)} \,|\, \mathcal{W} = i) = \exp\left(-\beta \left\| \mathbf{g}^{(q)} - \mathbf{w}_i \right\|^2\right) \tag{5.11}$$

Finally, the learning frequency of the neurons is utilized to compute the prior distribution $P(\mathcal{W} = i)$:

$$P(\mathcal{W} = i) = \frac{f_i^{(t)}}{\sum_{i'=1}^{N^{(t)}} f_{i'}^{(t)}} \tag{5.12}$$

Once the highly active neuron i^* for the query frame is known by Eq. (5.9), its activation strength is obtained as follows $a(i^*) = P(\mathcal{W} = i^* \,|\, I^{(q)})$. Hence, a location is predicted as novel, if $a(i^*) < \delta$. The curious reader might have noticed that $a(i^*)$ is the probability measure, and so it refers to the belief of the neuron in recalling the current location. If the neuron i^* fires with the probability greater than some certainty level (δ), the place is recognized as visited (i.e. a loop-closure detected).

5.4 Evaluation and Results

This section describes the benchmark datasets and the other system settings that are necessary for the experiments. Thereafter, the system's performance is evaluated on qualitative and quantitative grounds. Finally, the performance of the proposed system is compared with the popular place recognition methods using F-measure.

5.4.1 Experimental Setup

For experimental evaluation, three challenging datasets are considered, which include four St. Lucia sequences (Glover et al. 2010), two KITTI sequences (Geiger et al. 2012) and one sequence from the Oxford dataset (Cummins and Newman 2008). Figure 5.5 shows some images from the benchmark datasets. The images illustrate the visual complexity in natural environments. For example, observe the perceptual changes in appearance of the places due to traffic, dynamic lighting conditions, and the shadows of the trees. Amongst all the datasets, the St. Lucia is the longest one, as each sequence comprises 19−21K images for a 17 km long drive along the routes. These sequences comprise images from multiples times of the day and are recorded with a web camera operating at ∼15 Hz. The images have a resolution of 640×480 pixels. Regarding KITTI dataset, we utilized sequences 00 and 02, which consist of 4541 and 4661 images, respectively. These sequences are recorded at 10 Hz and the imagery has 1241×376 resolution. The CityCenter sequence of the Oxford dataset comes with the 1237 left-right camera image pairs, and we used the left camera images in experiments. The CityCenter sequence has the same resolution as St. Lucia. Table 5.1 summarizes the details of the datasets.

To evaluate the system's performance on CityCenter sequence, the ground truth matrix available from the authors is utilized. The algorithm's prediction at the query location is regarded as a true positive, false positive or negative by looking into the ground truth matrix. For the St. Lucia and KITTI sequences the GPS data provided with the datasets is availed, considering the translational error of 10 m and 4 m, respectively. No geometric information or visual odometry data is considered throughout the course of visual learning and place recognition. The algorithm is

(a) St. Lucia **(b)** KITTI **(c)** CityCenter

Figure 5.5: (a) Note the visual appearance of the same place at different times. The shown images belong to St. Lucia sequence SL08 (top) and SL14 (bottom). (b) In the middle column, top and bottom images represent the KITTI sequences 00 and 02, respectively. (c) For CityCenter, the noticeable changes in the scene are due to moving vehicles and lighting conditions along the sky.

Table 5.1: Details of the Benchmark Datasets

Dataset	Sequence (abbr.)	# Frames	Resolution (pixels)	Frame rate (Hz)	Distance (km)
St. Lucia	SL12	19251	640×480	15	17.6
	SL14	20894	640×480	15	17.6
	SL15	21434	640×480	15	17.6
	SL08	21815	640×480	15	17.6
Oxford	CC	1237	640×480	0.5	2.0
KITTI	KT00	4541	1241×376	10	3.7
	KT02	4661	1241×376	10	5.0

Following are the abbreviations for the datasets used in experiments (see details in text).
• SL12: 100909 (12:10) • SL14: 100909 (14:10) • SL15: 180809 (15:45) • SL08: 190809 (08:45)
• CC: CityCenter • KT00: KITTI Seq.# 00 • KT02: KITTI Seq.# 02

implemented in MATLAB that runs on a Linux machine with 3.4 GHz Intel Core i7 machine and 16 GB physical memory.

For all the experiments same parameter settings, as provided in Table 5.2, are used. The parameter configurations are selected empirically instead of fine tuning system's parameters for a specific sequence. We select those parameter values that reduce the root mean square error (RMSE) in the network.

Table 5.2: Standard Parameters Setup (Kazmi and Mertsching 2016b)

Parameter's Description	Value
Neuron instantiation weight (γ)	0.95
Learning rate (α)	0.1
Kernel's bandwidth (β)	64
Error tolerance of a neuron (τ)	0.1
Error smoothness factor (η)	0.99
Learning window size (ρ)	7

5.4.2 Correctness of the Learned Representation

To provide the quantitative evaluation of the feature maps learned by the algorithm, we compute the root mean square error (RMSE) in the network:

$$\epsilon(t) = \sqrt{\frac{1}{M}\sum_{i=1}^{N}\sum_{i' \in \omega_i}\left\|\mathbf{g}_{i'} - \mathbf{w}_i\right\|^2} \tag{5.13}$$

Note that for simplicity reasons, we have omitted the time superscript from the variables. Hence, M and N are the number frames retrieved and the number of

neurons in the network up to present time, respectively, where ω_i is the index set of the frames mapped to the neuron i. In practice, one utilizes Eq. (5.8) to obtain RMSE. The error profiles of the selected sequences are shown in Figure 5.6. One can thus comment that the learned representations are correct, if the network error decreases with time, especially over the re-visited routes. Notice that the network

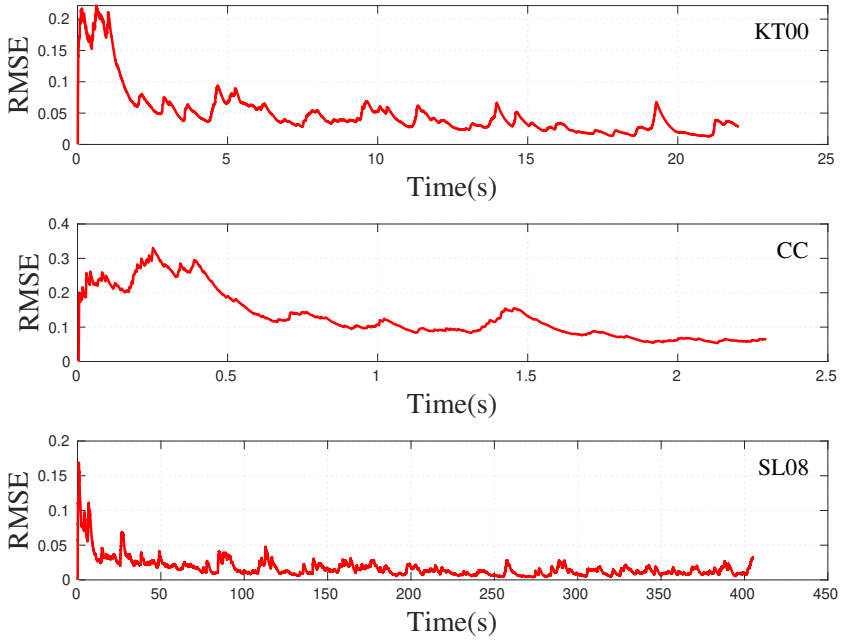


Figure 5.6: RMSE in the network for the selected sequences. Note that the error for the CC sequence remains relatively higher throughout the course of learning due to low frame rate. As the overlap between the frames is less, there is more information to learn between the frames (Kazmi and Mertsching 2016b).

started up with a large error across all the datasets, which is gradually regulated by the learning dynamics. The variations in error originate from those parts of environments where the learned representation significantly differs from current visual description of the scenes. This situation arises because of dynamic objects, appearance changes, or novel places.

5.4.3 Evolution of Neurons in the Network

To validate the correctness of the learned representation on qualitative grounds, the evolution of neurons in the network is examined. For this purpose, we selected SL08 sequence as a representative among the other sequences. The rational for this selection is that it is the longest among all sequences, has 24 intersection points and has numerous loop locations some of which are even traversed multiple times.

Figure 5.7 illustrates the continual growth of the network along the main diagonal. The progressive alignment of points below main diagonal refers to the selection of

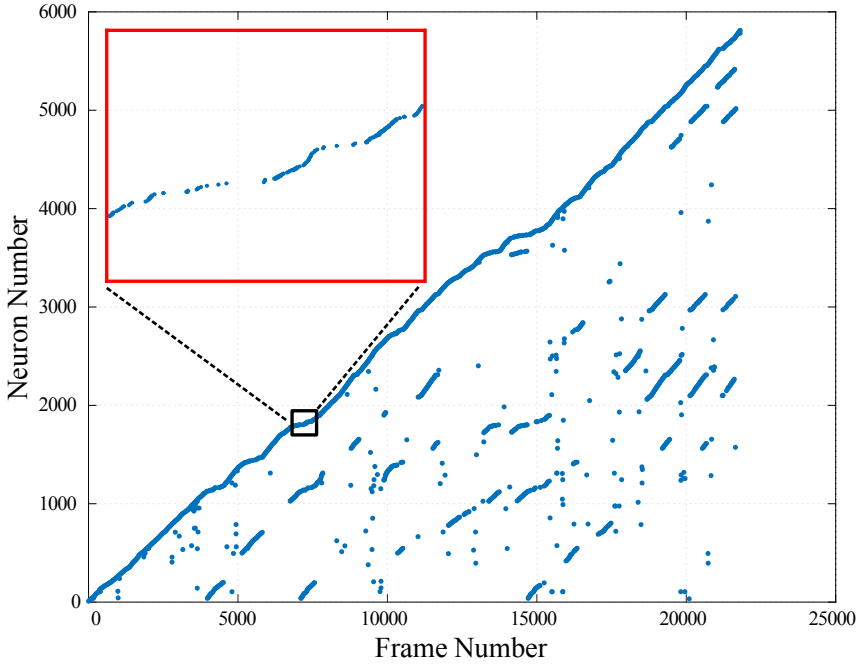


Figure 5.7: Each frame in the sequence is exclusively mapped to the nearest neuron. Note that the emerging off-diagonal recall curves do not reflect the place recognition. They are frame-to-neuron mappings – details in text. The creation of only few neurons in the magnified region demonstrates the ability of learning algorithm to compactly represent the environment (Kazmi and Mertsching 2016b).

the winner neurons for the given frames. Many similar constellations in the plot reflect the persistent utilization of the network. Note that unlike our previous work (chapter 4), here the selection of the winner neuron does not necessarily refer to a familiar location. In the current formulation, a winner is computed using Eq. (5.4), whereas the decision of place recognition is taken based on activation strength of the highly active neuron, given by Eq. (5.9). Determining a winner is the aspect of visual learning, while finding out the highly active neuron is the aspect of the recognition algorithm. So, one should not confuse the winner and highly active neurons in a conventional sense. Moreover, a recently created neuron can even be a winner neuron, as it is more likely for its weight to resemble the visual description of the current frame. By contrast, recently created neurons are not included in the decision of place recognition.

Within the zoomed region, one might have noticed the creation of few more neurons along the revisited route. Such a behavior reflects back to unavoidable external influences, e.g., appearance changes caused by shadows of the building or tree, lighting variations, and ample moving objects in the scenes. For instance, the network created 5807 neurons to learn 21,815 frames of SL08. To represent the same environment at a different day time (e.g. SL15), 5857 neurons are created. For KT00, the image-to-neuron ratio is 4541:1814, while in case of CC, the total created neurons are 726, which is almost the half of the frames in that sequence.

Figure 5.8 shows utilization of the network for the selected sequences. The activity profile of each neuron in the network refers to its learning frequency. Amongst all the sequences, the activity profile of the network for CC is lower because of large inter-frame translations. Moreover, CC has one loop closing route, which is also the reason why the network exhibits less utilization compared to other sequences (e.g. SL08 and KT00) that have multiple traversals through the loop locations.

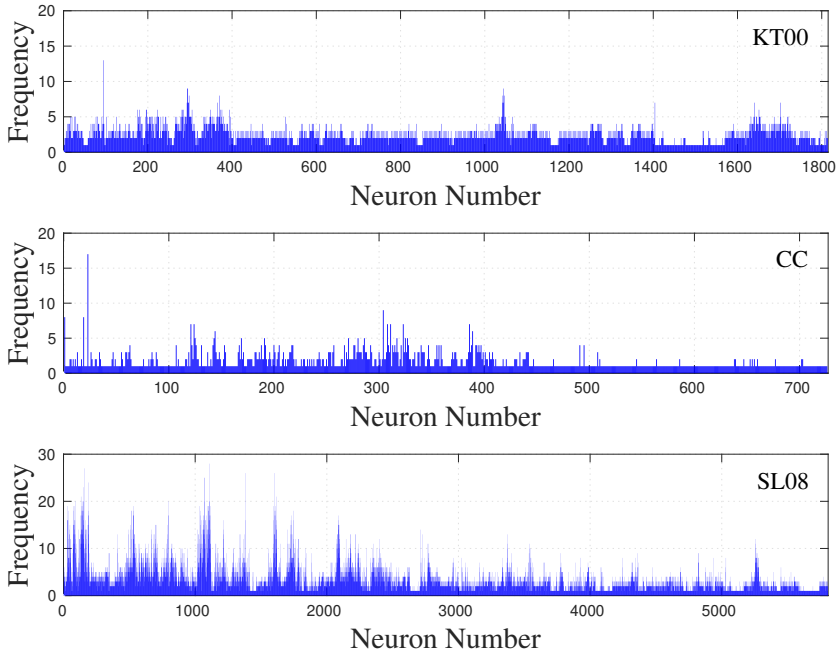


Figure 5.8: Heatmap of the network utilization. Frequency of a neuron refers to the number of frames for which it has been the winner.

5.4.4 Loop Detection Accuracy

This section evaluates the place recognition performance of the proposed algorithm in terms of precision–recall and F_1 scores, given by Eqs. (2.8), (2.9) and (2.10), respectively. Precision score demonstrates the tendency of the algorithm to make correct inferences, while recall refers to the algorithm's ability of not missing a correct inference. In formal terms, precision is the ratio of all correct inferences to all the inferences. On the other hand, recall is the proportion of correct inferences in all the inferences that are actually correct. With regard to the algorithm's inference, a query frame is predicted as familiar, if the activation strength $a(i^*)$ of the maximizer neuron i^* is greater than certain threshold δ value. Hence, the algorithm's prediction is evaluated as TP, if the query frame that is predicted familiar is actually familiar as per ground truth. Predicting a novel place as familiar is a FP case, and it is vice versa for FN.

To generate the precision-recall curves for the sequences, shown in Figure 5.9, we vary δ in the range $[0, 1]$. The obtained precision–recall results are interpretable in

terms of F_1-scores. On St. Lucia dataset, the highest F_1-score (88%) is achieved at SL08, given the precision of 91% and recall of 86%. In contrast, the maximum F_1-scores obtained for KT00 and CC sequences are 83% (precision: 84%, recall: 82%) and 75% (precision: 66%, recall: 87%), respectively. In Figure 5.10, we plot all the correct loop-closure detections for the best F_1-score achieved on the SL08 dataset. Notice the loop-closure performance of the proposed approach along re-visited routes (green circles). Moreover, most of the false negative cases happen on the turns due to viewpoint change. It is obvious to observe such a behavior for global descriptors (here gist), as they are viewpoint sensitive. The live demonstration of the proposed algorithm can be best visualized in the video (GET Lab 2016).

5.4.5 Discussion

Runtime complexity of the algorithm relates to the count of neurons in the network. For this purpose, SL08 being the largest test sequence is selected. The actual frame rate of the sequence is 15 Hz, whereas proposed algorithm processed the entire sequence at approx. 53 Hz. So, the mean processing time of a frame is 18.6 ms. Regarding KT00 and CC sequences, the mean processing times of our method are 4.9 ms and 1.9 ms, respectively. In addition, computation of gist features on average takes 94.8 ms per frame, including all the preprocessing overheads (e.g. image cropping, downsampling and patch normalizing).

We now bring about certain aspects to compare the place recognition performance of the proposed method (GBN) with the widely adopted baseline methods, namely FAB-MAP 2.0 (Cummins and Newman 2011) and SeqSLAM (Milford and Wyeth

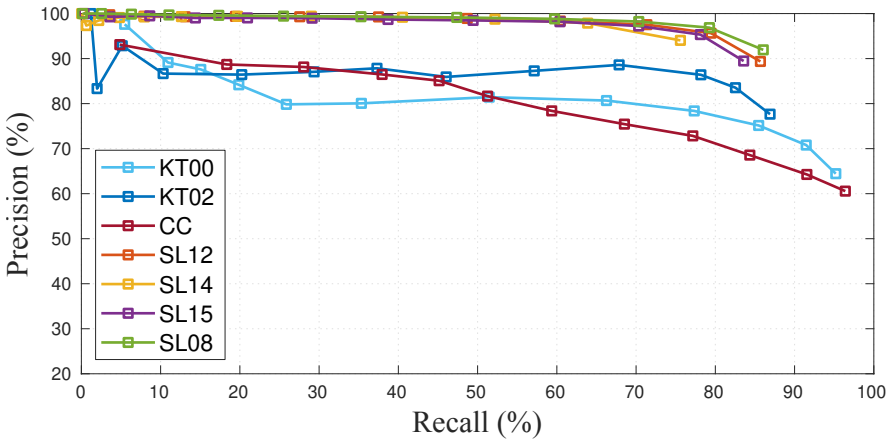


Figure 5.9: Notice the scale of the vertical axis. The proposed method maintains a significant balance between precision and recall scores. Even at higher recall levels there are no significant drops in precision. Achieving persistent precision for St. Lucia suburb ascribe to the high frame rate, which assists the network to settle down. Modified on (Kazmi and Mertsching 2016b).

2012). The SeqSLAM algorithm is tested using two different visual modalities, i.e., gist features and gray intensities of the down-sampled image. For experiments, binaries of FAB-MAP 2.0 are obtained from the authors and an open source implementation of SeqSLAM (Sünderhauf et al. 2013) is downloaded. The baseline algorithms have a list of parameters that make these approaches environment dependent. For instance, regarding FAB-MAP, we had to set the *false positive probability* to 0.02, whereas increasing it further significantly drops the precision scores. For the same reason, the sequence length parameter of SeqSLAM algorithm is set to 20.

The maximum F_1 scores achieved by the methods are used to evaluate their performance, as listed in Table 5.3. One might have noticed in the table that the methods exploiting global features, GBN (ours) and SeqSLAM, outperform the local features driven method (FAB-MAP 2.0). In general, GBN (ours) performs better than the baseline approaches, except for SeqSLAM (with gray features). One should, however, consider the fact that FAB-MAP 2.0 and GBN (ours) are single frame-based place recognition methods. By contrast, SeqSLAM use image sequences to detect loop locations and so it is obvious for the SeqSLAM to outperform due to more information. Despite such a significant difference, our algorithm showed an F_1-score of 88.9%, which is quite comparable to 89.87% score achieved by the gray variant of SeqSLAM. On the other hand, we outperform the gist-based SeqSLAM. Achieving a high F_1-score compared to gist-based variant of SeqSLAM show that the robust place recognition is not a coincidental choice of the feature space, rather it is the strength of our algorithm. The exhaustive experiments conducted on seven challenging sequences demonstrate the strength of our approach for real-time place recognition in unknown scenarios, without acquiring prior knowledge of the environment.

Table 5.3: Performance of the Well-known Place Recognition Approaches on SL08

Approach	**Features**	**Learning**	**Recognition Scheme**	F_1**-score** (%)
GBN (ours)	Gist	Online	MAP	88.9
SeqSLAM‡	Gist	–	DTW	85.82
SeqSLAM‡	Gray	–	DTW	89.87
FAB-MAP 2.0†	SURF	Offline	Bayes Filter	70.11

‡ Milford and Wyeth (2012) † Cummins and Newman (2011) – No learning procedure is followed. It is rather a matching technique. Note that variants of SeqSLAM follow different procedures to preprocess the image. E.g. Gray variant uses downsampled version of the whole image as a scene description. By contrast, for Gist computation we crop the image to 256×256 pixels which is certainly an information loss at the image boundaries. This essentially explains the performance reduction for Gist variant to handle environment complexity. Furthermore, a condition to find a minimum scoring trajectory degenerates the performance.

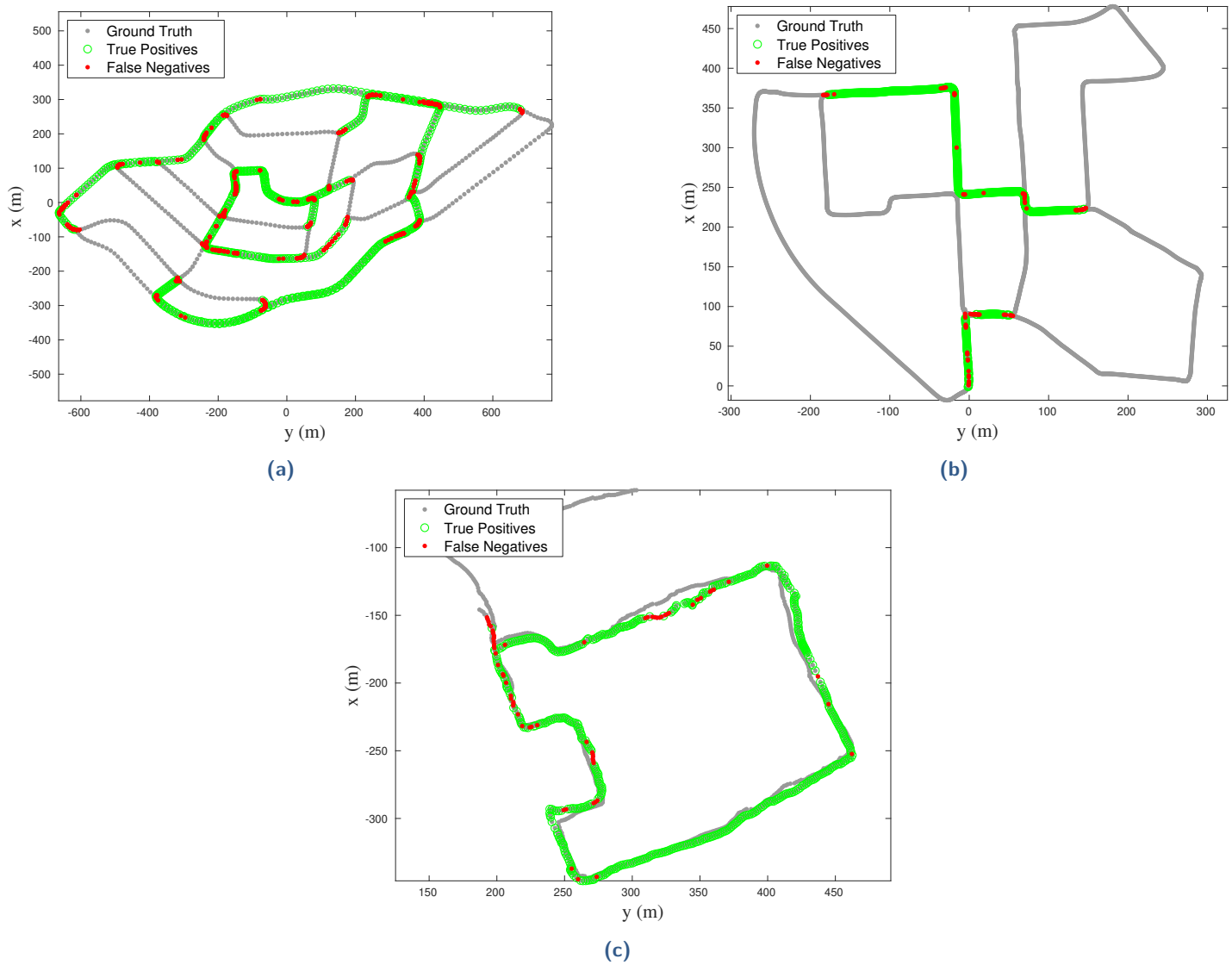


Figure 5.10: Loop detection results for (a) SL08 (b) KT00 and (c) CC sequences. Gray points are the location that are traversed only once. All green circles are the correct detection of the re-visited locations from our algorithm. A sequence of false negatives in CC (annotated red in range $-175 < x < -100$) occur on re-visiting the same route from an opposite direction. Our algorithm associates places in the network rather than place-to-place association. Hence, the trajectories (gray points) in vicinity of the visited routes (in CC) could not be mapped to each other.

5.5 Summary

This chapter addressed the challenges of lightweight real-time place recognition for appearance-based mapping. For this purpose, two major algorithms are introduced that perform simultaneous place learning and recognition. The framework is alto–gether named as growing belief network (GBN). Gist features are used to capture the visual representation of the scenes. To learn the extracted visual description, novel learning dynamics are proposed for GSOM that adapt the network to compensate the unexplained variance in the environment. Later, the incrementally trained network is utilized to obtained the maximum a posterior estimate over the neurons. Hence, each neuron can be treated as having the belief of representing the query place. The decision of the place recognition is taken, if the activation strength of the maximally active surpasses the predefined confidence level. The performance of the proposed approach is evaluated on seven sequences of varying challenges, such as dynamic object and dynamics lighting conditions. The experiments demonstrate that the robustness of the proposed to perform across the environment place recognition, without acquiring prior knowledge of the environment or offline training. In general, our algorithm has shown better performance than state-of-the-art methods. However, in certain cases, e.g., SeqSLAM (with gray features), our algorithm achieves slightly lower F_1 score, but it is because the baseline method uses more images to take the decision of place recognition rather than using a single frame.

6 Expectancy Detection of a Place through Nearby Context

In the preceding chapter, we introduced the concept of growing belief network (GBN), which offers a hierarchical solution to the online place recognition challenge. As natural environments are so filled with perceptual ambiguities, a single frame-based place recognition remains susceptible to false detections. Moreover, feature maps learned at a small constant learning rate under-represent the input space. As a result, the RMSE in the network retains for longer times. The present network dynamics are guided by the squared error metric to learn from input, thus weight normalization does not fit well with the learning rule. Besides this, ad-hoc parameter values insert new neurons at a fixed distance around the data point, ignoring the direction of error vector between the winner and the presented input.

With regard to the aforementioned problems, this chapter proposes several improvements in GBN to enhance the system's performance and further cut down the memory requirements. The contributions presented in the chapter primarily belong to our article published in IEEE Transactions on Robotics (Kazmi and Mertsching 2019a). Whereas, some aspects of the algorithm have appeared in the paper presented at International Conference on Computer Vision Systems (Kazmi et al. 2019b). The summary of the contributions is as follows: (a) We complement the spatial layout of a location with the visual appearance of the nearby places. (b) The decision of a loop detection is taken based on aggregated response of a series of highly active neurons, which we call *expected network activity*, as it is more reliable than observing the response of only a single highly active neuron. (c) Instead of using a constant learning rate criteria, we propose a decaying learning rate. The aim is to learn maximum in the first shot, and then gradually decrease the learning rate such that the neurons which have learned a massive proportion of the data freeze their positions in the feature space. (d) The learned network statistics are utilized to determine the insertion location of the new neuron. (e) Finally, we do not follow the weight normalization procedure in the present setup, and observe considerable improvement in computation time, memory requirements and the precision–recall performance of the algorithm.

Further discussion in this chapter proceeds as follows: Section 6.1 provides generic overview of the system's workflow in relation to the proposed improvements. Sec–

tion 6.2 introduces the reader to a novel approach for capturing the spatial layout of the scene. Improvements in the proposed learning and recognition algorithms are discussed in Sections 6.3 and 6.4, respectively. In Section 6.5, we provide an exhaustive evaluation of the proposed algorithms on four challenging datasets (com–prising eleven sequences). The section also includes the precision–recall comparison with the baseline methods, followed by their timing efficiency analysis. Later, the performance variation of the system is analyzed for various parameter configurations. In the end, Section 6.6 summarizes this chapter.

6.1 Introduction

This section briefly touches upon the working anatomy of the improved growing belief network (IGBN). The details of the specific improvements will become apparent as the discussion proceeds in the subsequent sections. Unlike, GBN, which provides a single frame-based visual description of a location, we define the visual description of a location as a weighted average of the gist features of the current and the preceding locations. For this purpose, we utilize the exponential smoothing, which puts more weights on the gist features of the most recent locations, while the weights drop exponentially for the locations that are father away. This procedure of feature extraction is illustrated in Figure 6.1.

The new visual description of the current location, denoted as $\bar{\mathbf{g}}^{(t)}$, is then fed to the visual learning and place recognition pipelines. Recognizing places based on a single maximizer neuron may result in false detection. To strengthen the decision making, IGBN observes the expected activity of the network, which is obtained from

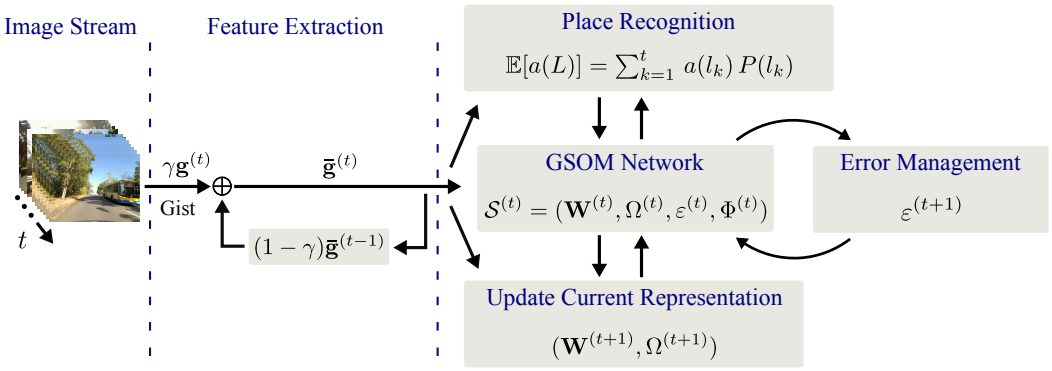


Figure 6.1: Improved growing belief network (IGBN): At a time instant t, the system grabs an image and computes its visual descriptor according to new definition. The extracted description of the current location is then utilized for visual learning and place recognition. The representation learned by the network is updated to account for the quantization error due to the query frame (Kazmi and Mertsching 2019a).

the averaged activation strength of the sequence of maximizer neurons; where the probability distribution $P(l_k)$ weighs the contribution of the maximizer neuron l_k to the total network activity. The algorithmic steps of the aforementioned procedure are listed in Algorithm 2.

Algorithm 2 Improved Growing Belief Network (IGBN) (Kazmi and Mertsching 2019a)

$\Omega^{(0)} \leftarrow 0$, $\varepsilon^{(0)} \leftarrow 0$
$\Phi^{(0)} \leftarrow$ initialize_params() ▷ Table 6.2
$\mathbf{W}^{(0)} \leftarrow$ initialize_weight()
while more images exist **do**
    $I^{(t)} \leftarrow$ get_image()
    $\mathbf{g}^{(t)} \leftarrow$ extract_gist_features($I^{(t)}$)
    $\bar{\mathbf{g}}^{(t)} \leftarrow$ obtain_spatial_layout($\bar{\mathbf{g}}^{(t-1)}, \mathbf{g}^{(t)}$) ▷ Eq. (6.1)
    GSOM_LEARNING($\mathcal{S}^{(t)}, \bar{\mathbf{g}}^{(t)}$)
    if $t \bmod \rho == 0$ **then**
        prune_inactive_neurons()
    end if
    $(E, l_t) =$ LOOP_CLOSURE_EXP($\mathcal{S}^{(t)}, \bar{\mathbf{g}}^{(t)}$)
    if $E > \delta$ **then**
        associate_locations($l_t, \Omega^{(t)}$) ▷ Loop detected
    else
        do_nothing() ▷ No loop detected
    end if
end while

6.2 Capturing the Spatial Layout of Scenes

In previous chapters, we described perceptual layout of a scene using its gist features. Such a description is not robust to perceptual aliasing due to high visual ambiguity in environments. Thus, we propose a novel scheme to capture the spatial layout of a scene. Instead of solely relying on the gist of the current location, we complement the visual description of current location with the gist of its preceding neighbors by means of exponential smoothing:

$$\bar{\mathbf{g}}^{(t)} = \gamma \mathbf{g}^{(t)} + (1-\gamma)\bar{\mathbf{g}}^{(t-1)} \tag{6.1}$$

Where $\bar{\mathbf{g}}^{(t)}$ is the new visual description of the current location. One might have noticed that the new description of the scene is the weighted average of gist features of the current $\mathbf{g}^{(t)}$ and the previous frames $\bar{\mathbf{g}}^{(t-1)}$. The parameter $\gamma \in (0, 1)$ is the smoothing factor, it determines the smoothness between the current and past observations. Expanding the right-hand side of the above equation results in:

$$\bar{\mathbf{g}}^{(t)} = \gamma \mathbf{g}^{(t)} + \gamma(1-\gamma)\mathbf{g}^{(t-1)} - (1-\gamma)^2 \bar{\mathbf{g}}^{(t-2)}$$

Note that, as the time passes by, the weight decay over the past observations becomes exponential. Hence, the recent observations contribute more to describe the current location than those that are farther away. In this regard, the choice of a suitable γ value is crucial. Too small gamma values pick up the frequency information of the farthest observations, whereas large values (close to 1) include the frequency content of the very recent locations. Rather than empirically deciding on the value of γ, we relate it to the learning epoch ρ and the frame rate R_{fps}:

$$\gamma = 1 - \exp\left(\frac{-\rho}{\max(\rho,\, R_{fps})}\right) \tag{6.2}$$

At high frame rates, the above expression assigns small values to γ, i.e., more smoothness. Such a choice is natural in case of a high frame rate because of gradual change of details between the frames. On the contrary, a low frame rate selects high γ and applies less smoothness due to rapid change of details. The denominator in the exponent ensures that the smoothing factor does not exceed one. Note that frame rate is related to the minimum age of the scene's statistics. For example, in the present setup it is $1/\gamma \approx 1.58$ frames, provided $R_{fps} \leq \rho$. Generally, one uses high smoothness level to include many observations in the weighted average. Nevertheless, too much smoothness influences the loop detection latency.

6.3 Improved Learning Dynamics for GSOM

The network dynamics of IGBN are described with the same set of state variables, as given in Eq. (5.1). The exception is for the parameters set $\Phi^{(t)}$, which is now indexed by the time t due to a decaying learning rate:

$$\mathcal{S}^{(t)} = \left(\mathbf{W}^{(t)},\, \Omega^{(t)},\, \varepsilon^{(t)},\, \Phi^{(t)}\right) \tag{6.3}$$

Regarding the definitions of the network variables and the description of the contents of a neuron, we refer to Section 5.2. Note that in the improved setup, the learning rate of a neuron is no more a constant quantity. Hence, we re-state the decaying learning rate of the i^{th} neuron as $0 < \alpha_i^{(k)} \leq \alpha_0$, where $a_0 < 1$ refers to the initial learning rate of the neuron. The following discussion describes the learning dynamics, including importance of network parameters, whereas the performance variation of the algorithm for various parameter configurations is provided in Section 6.5.5.

Network Initialization

In the IGBN, the initial size of the network is only one neuron, whereas the weight vector of the neuron are initialized according to Eq. (6.1). The initial condition in

the equation is set to $\bar{\mathbf{g}}^{(0)} = \mathbf{0}^\top$. Thus, one can follow that the weights of the neuron are sampled from the gist descriptor of the first frame. Indeed, the choice of a null vector in initial condition works better than a random initialization, as a random assignment degenerates the frequency statistics of the subsequent locations.

Determine the Winner Neuron

The matching rule for the winner search is given by Eq. (5.4). Now that we have a new description of the location $\bar{\mathbf{g}}^{(t)}$, the equation is re-expressed as:

$$c = \arg\min_i \left\| \bar{\mathbf{g}}^{(t)} - \mathbf{w}_i^{(t)} \right\|^2 \tag{6.4}$$

In the same vein, Eq. (5.5) for place-to-neuron mappings can be re-stated as:

$$\omega_c^{(t)} = \left\{ \bar{\mathbf{g}}^{(k)} : \left\| \bar{\mathbf{g}}^{(k)} - \mathbf{w}_c^{(t)} \right\|^2 \leq \left\| \bar{\mathbf{g}}^{(k)} - \mathbf{w}_i^{(t)} \right\|^2 \right\} \tag{6.5}$$

The above expression exclusively assigns places to the neurons. In practice, the places are incrementally mapped to the closest neuron. Hence, one only updates the assignments of the winner. Here, it is worth mentioning that the places with a common winner are not necessarily the physical neighbors. Consider, for instance, the neurons depicted in Figure 6.2. Some of the places that are mapped to the

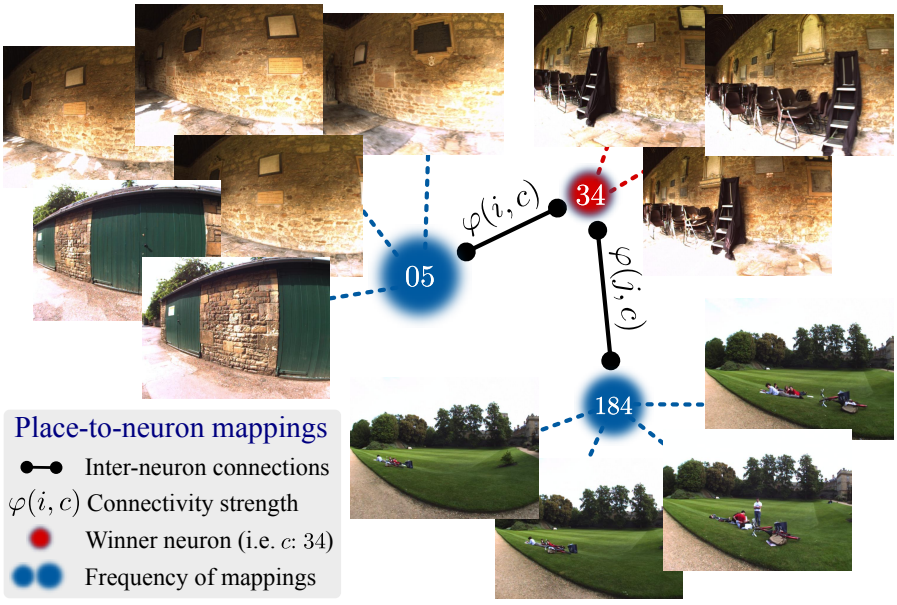


Figure 6.2: Place-to-neuron mappings: The shown neurons are the part of the network that incrementally learned the visual representation of the NewCollege environment (details in Section 6.5.1). The neuron's size indicates the learning frequency; bigger the neuron, more frequently it has been the winner. The neuron i: 05 is a topological neighbor of the winner c: 34, as $\varphi(i, c) \approx 0.04$. Whereas, neuron j: 184 receives zero-excitation from the winner (Kazmi and Mertsching 2019a).

neuron i: 05 look perceptually similar, but they are physically distant. Each frame for which c is the winner, increases the neuron's learning frequency $f_c^{(t+1)}$ by the increment of one.

Adapting the Network Weights

Unlike GBN, the weight update rule for IGBN is guided by the frequency sensitive learning criteria. In the improved network dynamics, the learning rate $\alpha_c^{(t)}$ is a decaying function, hence Eq. (5.6) is re-expressed as:

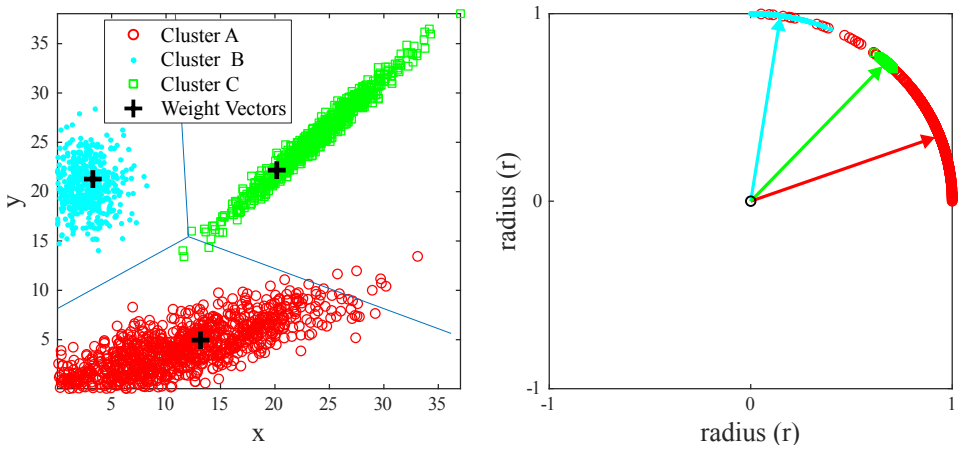
$$\mathbf{w}_i^{(t+1)} = \mathbf{w}_i^{(t)} + \alpha_c^{(t)} \varphi\left(\mathbf{w}_i^{(t)}, \mathbf{w}_c^{(t)}\right)\left(\bar{\mathbf{g}}^{(t)} - \mathbf{w}_i^{(t)}\right) \tag{6.6}$$

The kernel function $\varphi(.,.)$ is the Gaussian RBF centered at the winner. Follow further details in Section 5.2. Compared to the above weight update rule, the weight update in GBN relies on a constant learning rate, which keeps the network adaptive for an indefinite time. As the constant learning rates are usually set to small numerical values, one needs many iterations through the data to learn the representation of the underlying feature space. In online learning, training data appears sequentially and the inputs are discarded after adapting the network. Hence, re-iterating through the training examples is infeasible. By contrast, guessing a large constant learning rate introduces the overshooting problem in gradient descent, resulting in the relocation of neurons to even farthest positions from the current input. An appropriate strategy is to learn more from the input in the first shot, and then gradually drop the learning rate such that the frequently active neurons freeze their positions in the feature space. This saves the neurons from being pulled towards the noisy input or outliers. We update the learning rate of the winner as follows:

$$\alpha_c^{(t+1)} = \alpha_0 \frac{1}{f_c^{(t)}} \tag{6.7}$$

Note that the learning rate is a decaying function of the winner's learning frequency. After updating weights, usually the weight vectors are normalized to a unit length. As the present learning dynamics are backed by the squared error metric, the learned representation forms a Voronoi tessellation in the feature space. This concept is illustrated for a toy example in Figure 6.3a. Normalizing weights in the current setup does not payoff, as in the normalized space vectors are mapped to a unit circle, see Figure 6.3b, which is a unit sphere in higher dimensions. A suitable matching rule in the normalized space could be a cosine similarity. Generally, one must assure that the matching and weight update rules are mutually compliant (Kohonen 1990).

As data points lie on the surface of a high-dimensional sphere, the chances are that the inter-cluster boundaries become complicated, as it can be observed in Figure 6.3b.



(a) Unnormalized Space **(b)** Normalized Space

Figure 6.3: Toy example illustrating the effect of weight normalization in 2D. In the normalized, space the weight vectors are projected to a unit circle, their location on the unit circle is denoted by arrows. In a high-dimensional space this corresponds to points lying on the surface of a unit sphere. Using the squared-error metric for the normalized vectors is no more meaningful. A better matching rule in this case is a cosine similarity (follow details in text).

Although the effect of normalization in a high-dimensional space is quite smaller, the possibilities cannot be ignored. Indeed, to guarantee that the normalization does not influence the cluster boundaries in higher dimensions, one must ensure that the data points are uniformly distributed in the space – a condition that cannot be enforced in unknown scenarios. The weight normalization aspect of the system is evaluated in details in Section 6.5.6.

For IGBN, we do not follow the weight normalization procedure and notice (a) improved compression of the input space, (b) increased place recognition performance, and (c) the faster execution times due to the reduced size of the search space and relaxation from computing the square root.

Controlling the Network Growth

Analogous to Eq. (5.8), the quantization error due to current visual input is added to the error accumulator:

$$\varepsilon_c^{(t+1)} = \varepsilon_c^{(t)} + \sum_{j=1}^{D} \left(\bar{g}_j^{(t)} - w_{jc}^{(t)} \right)^2 \tag{6.8}$$

Creating a new neuron depends on the following conditions: (a) the quantization error of the winner exceeds the acceptable error tolerance, i.e., $\varepsilon_c^{(t+1)} > \tau$, or (b) the network is unable to represent the current location. This situation occurs when data point falls outside the convex hull of a neuron, given by (5.7).

Instantiating a new neuron $\mathbf{w}_r$ using Eq. (5.2) does not consider the direction of the error vector between the winner neuron and the input data points. As a result, the neuron is assigned a random location around the data point, as illustrated in Figure 6.4. Note that for both the points, i.e., $\mathbf{x}_1$ and $\mathbf{x}_2$, new locations of the neurons

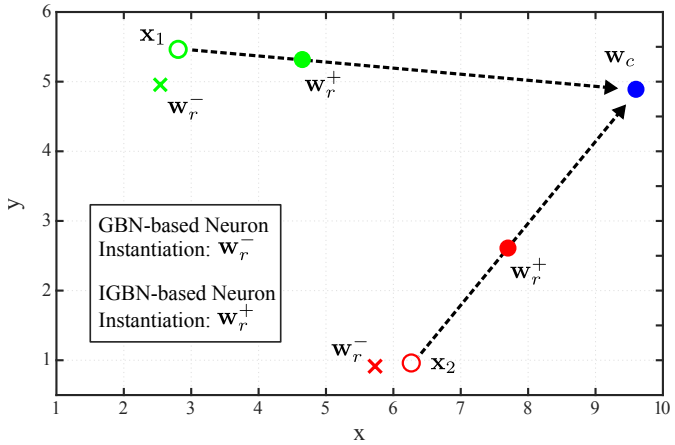


Figure 6.4: GBN vs. IGBN Neuron Instantiation (a 2D visualization). Suppose $\mathbf{x}_1$ and $\mathbf{x}_2$ are the two query points. Ideally, a new neuron must be instantiated along the direction of error vector between the winner and the query points. However, GBN randomly locates the new neuron ($\mathbf{w}_r^-$) near the data points. On the contrary, our new approach (IGBN) grows new neurons ($\mathbf{w}_r^+$) along the corresponding error vectors between winner and the query points. This effect also translates to the higher dimensions.

(i.e. $\mathbf{w}_r^-$) are even farther away from the winner. To fix this problem, we initialize the weights of the new neuron as a linear combination of the winner neuron's weight and the query descriptor:

$$\mathbf{w}_r^{(t)} = \frac{1}{\sigma + 1}\,\bar{\mathbf{g}}^{(t)} + \frac{\sigma}{\sigma + 1}\,\mathbf{w}_c^{(t)} \tag{6.9}$$

The scoring function $\sigma = P(I^{(t)} \,|\, \mathcal{W} = c)$ is calculated from Eq. (6.13). It is used to distribute the importance score between the descriptor and the winner's weights. The corresponding SSE for the new neuron is initialized as follows:

$$\varepsilon_r^{(t)} = \left(\bar{\mathbf{g}}^{(t)} - \mathbf{w}_r^{(t)}\right)^\top \left(\bar{\mathbf{g}}^{(t)} - \mathbf{w}_r^{(t)}\right) \tag{6.10}$$

Lastly, similar to GBN, the network error is smoothed by a tiny amount, which facilitates the network dynamics against the growth of too many neurons in the regions of high density.

Network Pruning

Network pruning is the procedure of removing spurious neurons from the network. Analogous to our previous approach (i.e. GBN), we perform network pruning after

each learning epoch, see definition in Section 5.2. In an earlier chapter, we already explained the steps for network pruning. Here, we comment on how the size of the learning epoch ρ influences the computational complexity of the algorithm. In case of a large learning epoch, the spurious neurons occupy the system's resources for longer times. The presence of un-utilized neurons in the network makes the winner search and neighborhood finding computationally intensive. On the other hand, if $\rho >> R_{fps}$, the same level of smoothness is applied to all the datasets, which does not agree with the smoothness condition we set earlier in Section 6.2.

An algorithmic description of the entire visual learning procedure is provided in Algorithm 3.

Algorithm 3 Visual Learning with GSOM (Kazmi and Mertsching 2019a)

procedure GSOM_LEARNING($\mathcal{S}^{(t)}, \bar{\mathbf{g}}^{(t)}$)
  $c \leftarrow$ find_winner_neuron($\mathbf{W}^{(t)}, \bar{\mathbf{g}}^{(t)}$) ▷ Eq. (6.4)
  $\mathbf{W}^{(t+1)} \leftarrow$ adapt_weights($\mathbf{W}^{(t)}, \bar{\mathbf{g}}^{(t)}, \Phi^{(t)}$) ▷ Eq. (6.6)
  $\omega_c^{(t+1)} \leftarrow$ new_mappings($\omega_c^{(t)}$, curr_frame_id) ▷ Eq. (6.5)
  $\alpha_c^{(t+1)} \leftarrow$ update_learning_rate($\alpha_0, f_c^{(t)}$) ▷ Eq. (6.7)
  $f_c^{(t+1)} \leftarrow f_c^{(t)} + 1$
  $\varepsilon_c^{(t+1)} \leftarrow$ error_update($\varepsilon_c^{(t)}, \bar{\mathbf{g}}^{(t)}$) ▷ Eq. (6.8)
  if $\varepsilon_c^{(t+1)} > \tau$ **then**
    grow_new_neuron() ▷ Eqs. (6.9) and (6.10)
  end if
  $\varepsilon^{(t+1)} = \eta\, \varepsilon^{(t)}$ ▷ Error smoothing
end procedure

6.4 Loop-closure Detection

Finding a single maximizer neuron for the query image may yield false positive detections. It is, therefore, appropriate to look out for the activation strength of a sequence of maximally active neurons. The fact that neurons are highly active for a sequence of images, strengthens the belief that vehicle is revisiting a previously visited route. Hence, to decide on a loop-closure event, we aggregate the activity of a sequence of maximizer neurons, which we call an *expected network activity*. This concept is illustrated in Figure 6.5.

Consider a discrete random variable L that instantiates to the maximizer neuron at a time instant k. A particular realization of the variable is denoted as l_k. Note that l_k is analogous to the maximizer neuron i^*, given by Eq. (5.9), in a single-frame based loop detection. In the present case, the maximizer neuron itself is treated as a random variable to observe the net response of the network for a sequence of image frames. If $a(L)$ is a function of the random variable L, denoting the activation

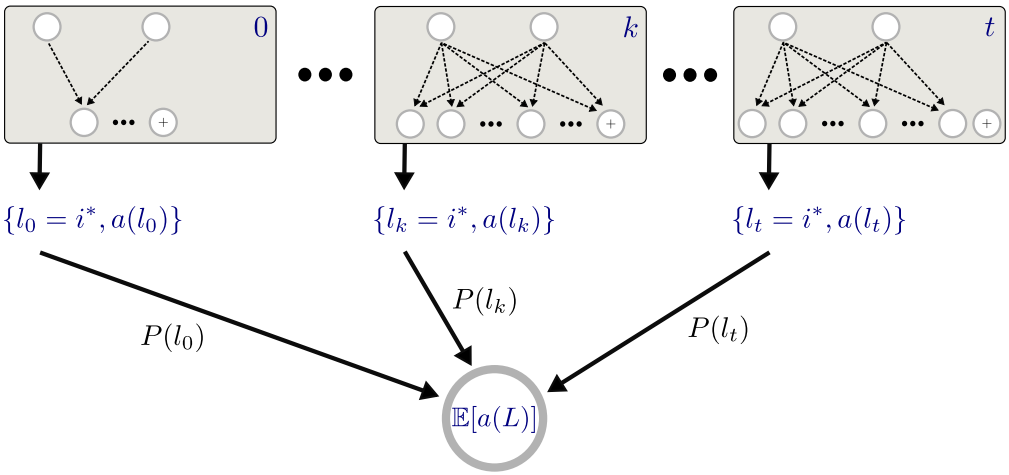


Figure 6.5: Proposed framework for improved growing belief network (IGBN). At each time instant k, the output of the belief network is the maximizer neuron l_k and its activation strength $a(l_k)$. The expected network activity is obtained from the activation strength of sequence of maximizer neurons. The probability distribution $P(.)$ decides on which connections are selected from the history.

strength of the maximizer neuron, then by the *law of unconscious statistician*, the expected activation strength of the network is given by:

$$\mathbb{E}[a(L)] = \sum_{k=1}^{t} a(l_k)\, P(l_k) \tag{6.11}$$

The probability distribution $P(.)$ over the random variable L is unknown. Indeed, real-world environments involve ample randomness. Thus, the probability of a neuron showing maximum activation can be assumed uncorrelated and equally likely in time. This behavior is modeled with the uniform distribution:

$$P(l_k) = \begin{cases} \frac{1}{t - t_0} & t_0 < k \leq t \\ 0 & \text{otherwise} \end{cases}$$

The parameter t_0 determines how far in time the maximally active neurons are observed. Considering a large history of the maximizer neurons, smooths out the expected network activity at the loop locations. Indeed, for a large $\Delta t = t - t_0$, one can discern that the no-loop locations start to appear in the sum of Eq. (6.11). Due to the fact that at no-loop cases the neurons' activation strength is low, the expected network's activity results in a drop-off. The activation strength of a maximizer neuron is its posterior probability, which we obtain as follows:

$$a(l_k) = \max_{i \in [1,\, N^{(k)} - \mu]} P(\mathcal{W} = i \,|\, I^{(k)}) \tag{6.12}$$

One might have noticed that the above expression is quite similar to Eq. (5.9), except that in the present case, we retrieve the maximum activation strength rather than

the neuron itself. The right-hand side of the equation can be solved using Bayes rule. For details, we refer the reader to Eq. (5.10). The likelihood and the prior probabilities in the expression are computed using equations (5.11) and (5.12), respectively. Here, we re-express the definition of likelihood due to the fact that we have new visual description of the query frame $I^{(k)}$ at a particular time k:

$$P(I^{(k)} \,|\, \mathcal{W} = i) = \exp\left(-\beta \left\| \bar{\mathbf{g}}^{(k)} - \mathbf{w}_i \right\|^2\right) \tag{6.13}$$

Finally, Eq. (6.11) is used to take the decision of a loop-closing event. For instance, given the query image, if the expected network activity exceeds the tolerable certainty level (δ), i.e., $\mathbb{E}[a(L)] > \delta$, the query location is classified as familiar. The algorithmic details of the loop-closing process are given in Algorithm 4.

Algorithm 4 Loop-closure Detection (Kazmi and Mertsching 2019a)

procedure LOOP_CLOSURE_EXP($\mathcal{S}^{(t)}, \bar{\mathbf{g}}^{(t)}$)
  $E \leftarrow 0, l_t \leftarrow 0$
  $\Delta t \leftarrow t - t_0$
  for $k \leftarrow t_0$ to t **do**
    $P(I^{(k)} \,|\, \mathcal{W}) \leftarrow$ compute_likelihood(k) ▷ (6.13)
    $P(\mathcal{W}) \leftarrow$ compute_prior_prob(k) ▷ (5.12)
    $P(\mathcal{W} \,|\, I^{(k)}) \leftarrow$ estimate_posterior_prob(k) ▷ (5.10)
    $(a_k, l_k) \leftarrow$ maximize_aposteriori() ▷ (6.12)
    if $k == t$ **then**
      $l_t \leftarrow l_k$
    end if
    $E \leftarrow E + \frac{1}{\Delta t} a_k$ ▷ (6.11)
  end for
  return E, l_t
end procedure

6.5 Evaluation and Results

This section first introduces the benchmark datasets that are used in experiments. Thereon, the system configurations and parameter settings are defined. We then evaluate our algorithm's precision–recall performance and the timing efficiency, and compare the results with the state-of-the-art. Finally, the performance variation analysis of the proposed approach is performed for different parameters, followed by a discussion on various system's aspects.

6.5.1 Experimental Setup

For experiments, eleven challenging sequences are selected from four different datasets of varying complexity level. The datasets include four sequences each from

the St. Lucia (Glover et al. 2010) and the KITTI (Geiger et al. 2012) environments, two sequences from the Oxford dataset (Cummins and Newman 2008), and one sequence from the Malaga parking area (Blanco et al. 2009). The description of the datasets is listed in Table 6.1.

Table 6.1: Details of the Benchmark Datasets

Dataset	**Sequence** (abbr.)	**# Frames**	**Resolution** (pixels)	**Frame rate** (Hz)	**Distance** (km)
St. Lucia	SL12	19251	640×480	15	17.6
	SL14	20894	640×480	15	17.6
	SL15	21434	640×480	15	17.6
	SL08	21815	640×480	15	17.6
Oxford	CC	1237	640×480	0.5	2.0
	NC	1073	640×480	0.5	1.9
KITTI	KT00	4541	1241×376	10	3.7
	KT02	4661	1241×376	10	5.0
	KT05	2761	1241×376	10	2.2
	KT06	1101	1241×376	10	1.2
Malaga	MP6L	3474	1024×768	7.5	1.2

• SL12: 100909 (12:10) • SL14: 100909 (14:10) • SL15: 180809 (15:45) • SL08: 190809 (08:45)
• CC: CityCenter • NC: NewCollege • KT00: KITTI Seq.# 00 • KT02: KITTI Seq.# 02
• KT05: KITTI Seq.# 05 • KT06: KITTI Seq.# 06• MP6L: Malaga Parking 6L

Amongst all sequences, the St. Lucia sequences are longest traversed routes with 24 intersections. The Malaga dataset comes with stereo images, but we consider only the left camera images in the experiments. The Oxford datasets, i.e. NewCollege and CityCenter, are recorded with the lateral cameras pointing sideways. We utilize only the right camera images out of left-right image pair. The NewCollege dataset is challenging among other, as it involves perceptual similarities arising from a garden area and archways. For St. Lucia and Malaga environments, the provided GPS logs are availed as a ground truth, considering the GPS localization error of 10 m and 4 m, respectively. Moreover, in the Malaga dataset, similar to (Mur-Artal and Tardós 2014), the bi-directional traverses along the routes are not evaluated. Regarding the Oxford datasets, corresponding ground truth matrices are utilized for evaluating the system's loop-closure performance. Whereas, the ground truth available from (Arroyo et al. 2014) is employed for evaluation on the KITTI datasets.

Consistent with our previous experiments, i.e., Section 5.4.1, we do not use the geometric data or visual odometry during the entire course of visual learning and place recognition. The algorithm's implementation is done in MATLAB, while all the experiments are executed on a Linux machine with 16 GB physical memory and a 3.4 GHz Intel Core i7 processor. Instead of manually tuning the network parameters, the optimal configurations, listed in Table 6.2, are obtained through a grid search

Table 6.2: Standard Parameter Settings (Kazmi and Mertsching 2019a)

Parameter's Description	Value
Initial learning rate (α_0)	0.3
Kernel's bandwidth (β)	64
Error tolerance of a neuron (τ)	0.1
Error smoothness factor (η)	0.99
Learning window size (ρ)	7

approach. For this purpose, St. Lucia looptest sequence is utilized and those settings are selected that reduce the RMSE in the network and maximize the loop detection performance of the algorithm.

6.5.2 Loop Detection Accuracy

This section evaluates the place recognition performance of the proposed algorithm in terms of precision–recall (PR) scores, which are calculated using Eqs. (2.8) and (2.9). A query location is regarded as a loop or no-loop location based on expected network activity, given by Eq. (6.11). Hence, the algorithm's decision at any location is evaluated as TP, if the algorithm predicts the location as a loop location and it is also a loop location according to the ground truth. Predicting a no-loop location as a loop location is a FP detection, and it is vice versa for FN. To generate the PR-curves across the datasets, we vary the threshold δ in the range $(0, 1]$. The obtained results are plotted in Figure 6.6. In the plot, we do not include the scores for $\delta < 0.06$. Below the mentioned threshold value, the precision scores are not worthwhile for the loop detection task.

For autonomous navigation, a high precision score is very important, particularly when a metric mapping is needed. A single false detection may otherwise induce inconsistencies in the map and thus leading to a catastrophic failure of the whole system. Hence, we observe the maximum recalls achieved by our approach at 100% precision, see the results overlaid as a bar graph in Figure 6.6. The figure illustrates that IGBN successfully detects the loop locations between 51-97% of the times. Among all benchmark sequences, KT06 is the smallest and it possesses only a single loop closing route. Thus, highest recall rate, i.e., ca. 97%, in this particular case is not an unexpected performance of the algorithm. It is rather worth considering the performance of IGBN in St. Lucia environment, which indeed has many loop closing routes and even several of them are traversed multiple times.

For further experiments, we choose a set of four sequences, namely SL08, CC, KT00, and MP6L, as a representative of the datasets. Figure 6.7 shows some images of

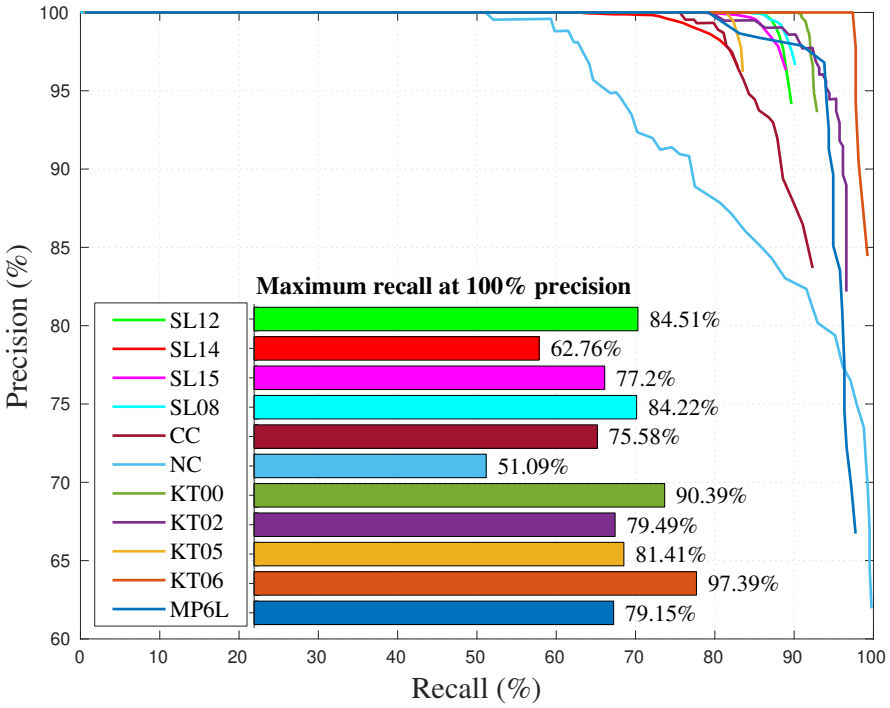


Figure 6.6: Notice the scale along the vertical axis. IGBN shows high precision–recall scores across the datasets. The lowest recalls at 100% precision are 62.76% and 51.09%, which belong to SL14 and NC, respectively. We noticed that SL14 is challenging among others because of the sun blaze and several shadows along the routes. The imagery of NC has ample visual ambiguities due the laterally tilted camera. Besides this, the garden and resembling man-made structures cause severe perceptual aliasing (Kazmi and Mertsching 2019a).

the loop locations from the representative set, which are correctly detected by our algorithm. Note the extreme brightness due to sun flare and shadows on the road in the query and match frames in Figure 6.7a. In CC sequence, i.e., Figure 6.7b, the layout of the scene is partially covered by a lorry. Moreover, the widespread shadows of the trees gradually disappear from the track. Despite such a severe operating conditions, our algorithm detects the correct matches. On the other hand, Figure 6.7d reflects the robust place recognition performance of our algorithm in a textureless scenario.

Figure 6.7: Some loop locations from representative datasets that are correctly recognized by IGBN. The second row refers to the closest match to the query image (first row) in a set of places associated with the neuron.

In Figure 6.8, we depict all the correct loop detections of our algorithm for the representative test sets. The correct detections are plotted as an overlay of green points over the vehicle's trajectory. The plots are consistent with the highest recalls that our algorithm achieves at 100% precision. There are no false positive matches at hundred percent precision criteria. In KT00 environment, the area to the left of the zoomed window is the part of the route that is not recognized by our algorithm. At the first traversal, the vehicle approaches the junction from the south direction, whereas the same track is revisited from the north side. Entering the same route from the opposite sides of the same junction, results in a severe viewpoint change. Such viewpoint changes cannot be handled by the global descriptors. Besides this, the exponential smoothing influences the loop detection latency to some degree.

The conventional appearance-based methods, e.g., (Cummins and Newman 2011), perform direct place-to-place association. However, our algorithm maps places to the nearest neuron. Thereby, we do not enforce one-to-one correspondence between places. In our algorithm, a query location may be associated to two (or more) reference locations, i.e., places that are learned by the network. For example, notice the red links between the locations in the zoomed windows across the test sequences. Several query locations have multiple associations in the network.

Following loop detection, the place association can be guided by several schemes or a combination of them. For instance, a simplest approach is a descriptor matching. In this case, the global descriptor of the query frame is matched with the descriptors of the places mapped to the maximizer neurons. Lastly, the location with the least perceptual distance is selected. Another workaround is to use of the GPS or odometry priors to guide the association. However, a more sophisticated approach is to extract the local features from the query frames and the set of frames assigned to the maximizer neurons. Thereupon, one performs the geometric consistency test between the local features, similar to the work of Mur-Artal and Tardós (2014). Hence, the frames with the least re-projection error are deemed the best association. In scenarios where metric mapping is desirable, such an approach will complement the filtering or pose-graph optimization algorithms to reduce the drifts in the maps.

To visualize the place associations in Figure 6.8, we proceed as follows: the places mapped to the maximizer neurons are retrieved and their distances to the query frame are computed. Finally, the association between the places with minimum GPS distance are drawn in the plot. One should, however, note that the GPS signal is adopted only for the visualization reasons in the figure. The result produced in Figure 6.6 and Table 6.3 neither use GPS nor any other sort of geometric data.

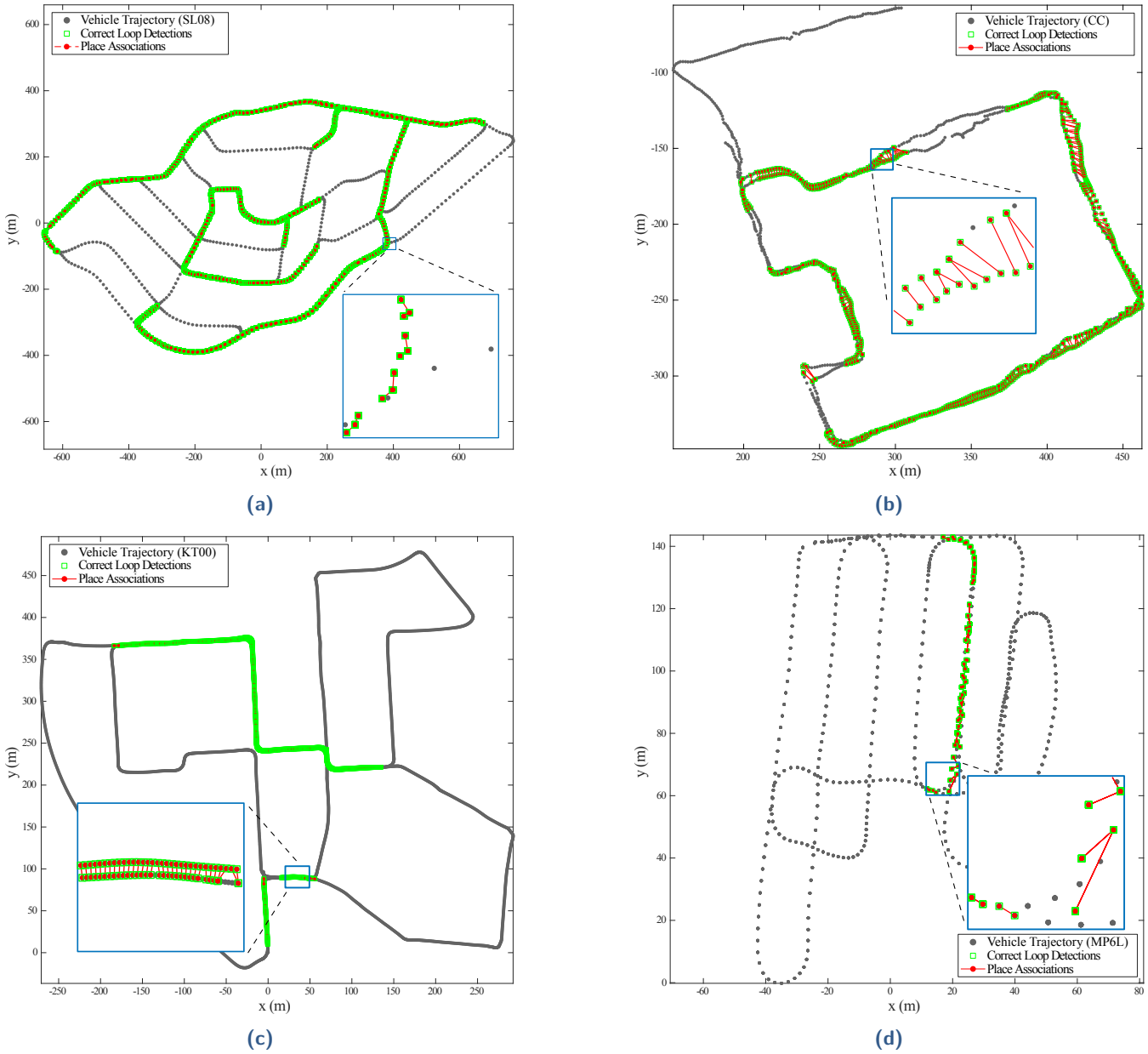


Figure 6.8: Loop detection results for the representative test sequences. The gray trajectory points are the locations that are visited only once. As we observe the maximum recalls at 100% precision, there are no false positives on gray routes. Green circles belong to the correct detections over the re-visited routes (Kazmi and Mertsching 2019a).

6.5.3 Comparison with the Baseline Techniques

This sections compares the performance of the proposed approach, namely IGBN, with the baseline methods. For this purpose, the maximums recalls achieved by the baseline algorithms (at 100% precision) are observed across the benchmark datasets. A list of baseline techniques includes three state-of-the-art approaches, i.e., FAB-MAP 2.0 (Cummins and Newman 2011), SeqSLAM (Milford and Wyeth 2012) and iBoW-LCD (Garcia-Fidalgo and Ortiz 2018). The performance of SeqSLAM is tested for three different visual modalities: gray intensities of the down-sampled image (SeqSLAM+gray), gist features (SeqSLAM+gist), and the features of the Conv3 layer obtained from a pre-trained Places205 convolutional network (Zhou et al. 2014) (SeqSLAM+CNN). Besides this, we consider the comparison with our previous work, i.e., GBN (Kazmi and Mertsching 2016b).

For experiments, binaries of the FAB-MAP 2.0 are obtained from the authors. Regarding SeqSLAM, the open source implementation available from (Sünderhauf et al. 2013) is downloaded. With regard to iBoW-LCD, the online code available from the authors (Garcia-Fidalgo and Ortiz 2018) is utilized. The evaluation of FAB-MAP and iBoW-LCD is performed with the default parameter values as provided in the code. To assure 100% precision for the FAB-MAP at $p \geq 0.99$, we set the *false positive probability* in the range 0.01 and 0.07, subject to the test sequence. In addition, rather than generating a new vocabulary, we utilize a pre-trained visual vocabulary (Cummins and Newman 2008). The parameters of SeqSLAM (Milford and Wyeth 2012) and GBN (Kazmi and Mertsching 2016b) are configured as stated in the respective works. The choice of a suitable sequence length parameter is crucial to the SeqSLAM's performance. Because we are interested in the recalls at 100% precision, the sequence length is configured to 20 frames. Increasing the length beyond this value drops the precision scores at some sequences, such as KITTI.

In the experiments, all the sequences are processed at their original frame rates, which are listed in Table 6.1. However, the exception is for the St. Lucia environment, which is processed at 4 Hz to reduce the inter-frame overlaps. Otherwise, the precision scores of the baseline approaches significantly degenerate. As a result, the baseline methods never show high recalls for the 100% precision criteria. Table 6.3 enlists the highest recall rates of our algorithm and the baseline approaches at 100% precision. The maximum recalls scored by our algorithm are recorded at $\delta = 0.3$. Unlike in the baseline methods, esp. FAB-MAP and SeqSLAM, we do not adjust the parameters according to environment. This demonstrate the general applicability of the proposed approach in diverse environments.

Table 6.3: Maximum Recalls at 100% Precision: Baseline Methods vs. Proposed Approach (Kazmi and Mertsching 2019a)

Test Sequence	FAB-MAP 2.0†	SeqSLAM+gray‡	SeqSLAM+gist‡	SeqSLAM+CNN‡	iBoW-LCD¶	GBN∼	IGBN+
SL12	57.07	65.54	61.87	67.11	21.26	1.81	**80.06**
SL14	35.17	51.98	54.32	54.93	52.15	0.04	**58.10**
SL15	46.30	64.79	56.09	57.42	36.87	0.31	**72.55**
SL08	53.98	67.96	63.29	60.08	41.32	4.22	**80.13**
CC	40.11	66.29	75.12	46.67	75.4	–	**75.58**
NC	52.63	46.15	**66.67**	47.62	53.03	–	51.09*
KT00	61.22	68.86	78.33	86.91	76.50	–	**90.39**
KT02	44.34	52.05	72.0	76.56	72.22	1.19	**79.49**
KT05	48.51	54.89	61.48	66.01	53.07	–	**81.41**
KT06	64.55	76.85	62.22	45.37	95.53	–	**97.39**
MP6L	21.83	45.37	35.58	31.12	46.36	–	**50.98**

† Regarding the FAB-MAP approach (Cummins and Newman 2011), some works, e.g., (Glover et al. 2010; Sünderhauf et al. 2015b) report that this approach does not address the challenges of St. Lucia environment. Our experiments with FAB-MAP show higher recalls across the datasets than those reported by Mur-Artal and Tardós (2014) and Garcia-Fidalgo and Ortiz (2017). On the MP6L sequence, Mur-Artal and Tardós (2014) state different recall rates for FAB-MAP than those listed in this table. It is because they processed the sequence at a very low frame rate (i.e. 1 Hz). ‡ SeqSLAM (Milford and Wyeth 2012) is a sequence matching method. It does not follow any learning procedure. ¶ The recalls listed for KT00 and KT06 are obtained from the original work (Garcia-Fidalgo and Ortiz 2018). Regarding CC and NC, our recalls differ from (Garcia-Fidalgo and Ortiz 2018), because we considered only the right camera images in evaluation. ∼ Our previous approach (Kazmi and Mertsching 2016b) does not show 100% precision on majority of the sequences. In general, the recalls remain less than 5% for the cases where 100% precision is observed. – The 100% precision was never reached on these datasets. ∗ On this sequence, IGBN shows relatively low performance. This sequence is recorded with laterally tilted camera. So, there is a large viewpoint factor to handle. For instance, while traversing the corridor, the camera mounted on the vehicle always confronts the wall. The lack of visual information introduces visual ambiguities at multiple locations. Moreover, the resembling man-made structures along the circular track give an impression of a familiar location.

Clearly, our new approach, i.e., IGBN, outperforms the baseline methods on almost all the sequences. Although the algorithm does not show superior performance on the NC sequence, the results are still comparable to the FAB-MAP and iBoW-LCD algorithms. SeqSLAM+gist shows the best performance on this test sequence. Note that on the Oxford dataset, gist-based variant of SeqSLAM consistently outperforms the other variants. We noticed that SeqSLAM+gray outperformed its gist counterpart on several test sequences. This could be attributed to the way in which SeqSLAM algorithm exploits features to find the loop locations. Among the vocabulary-based methods, which include iBoW-LCD and FAB-MAP, the former performs better nearly on all the test sequences. In St. Lucia environment, iBoW-LCD encounters a consid–erate amount of performance degradation due to volatile lighting conditions and multiple traversal through the routes. Although the authors report that iBoW-LCD is feature-agnostic (Garcia-Fidalgo and Ortiz 2018), it is only tested for the binary features; whereas binary features are extremely sensitive to the illumination changes in the scenes.

The gist-based variant of SeqSLAM shows comparable performance to the SeqS–LAM+CNN. It is not surprising to observe the gist features outperforming CNN. Indeed, in several situations the gist-based image representation has outperformed the CNN-based scene representations (Chen et al. 2015; Hou et al. 2015). How–ever, it is worth noticing that compared to gist features, CNN features are robust to illumination changes. This behavior can observed for SeqSLAM+CNN on the St. Lucia sequences that are recorded in afternoon, i.e., SL12, SL14 and SL15. By contrast, the CNN-based variant of SeqSLAM performs poorly on the Oxford and Malaga, possibly because of ample viewpoints changes in these datasets. The Conv3 features of the CNN are not robust to such a variation. Indeed, the choice of features is not the only aspect that influences the loop detection performance. Rather the learning and recognition algorithms also have a significant impact on the overall place recognition performance. It can be noticed in the table that on several test environments, SeqSLAM+gist has low recalls, whereas our approach has still scored higher recalls in those cases despite using the gist features.

6.5.4 Time Requirements

This section evaluates the computational requirements of the proposed algorithm on the representative test set. For this purpose, the timings for learning and recognizing a query image are recorded individually. This time log is plotted as a function of the frame number in Figure 6.9. Note that the learning and total execution times escalate as more frames are retrieved from the sequences. By contrast, the loop detection timings across the sequences do not increase by a large amount. Clearly, most of the computational load is on the shoulder of the learning algorithm. For

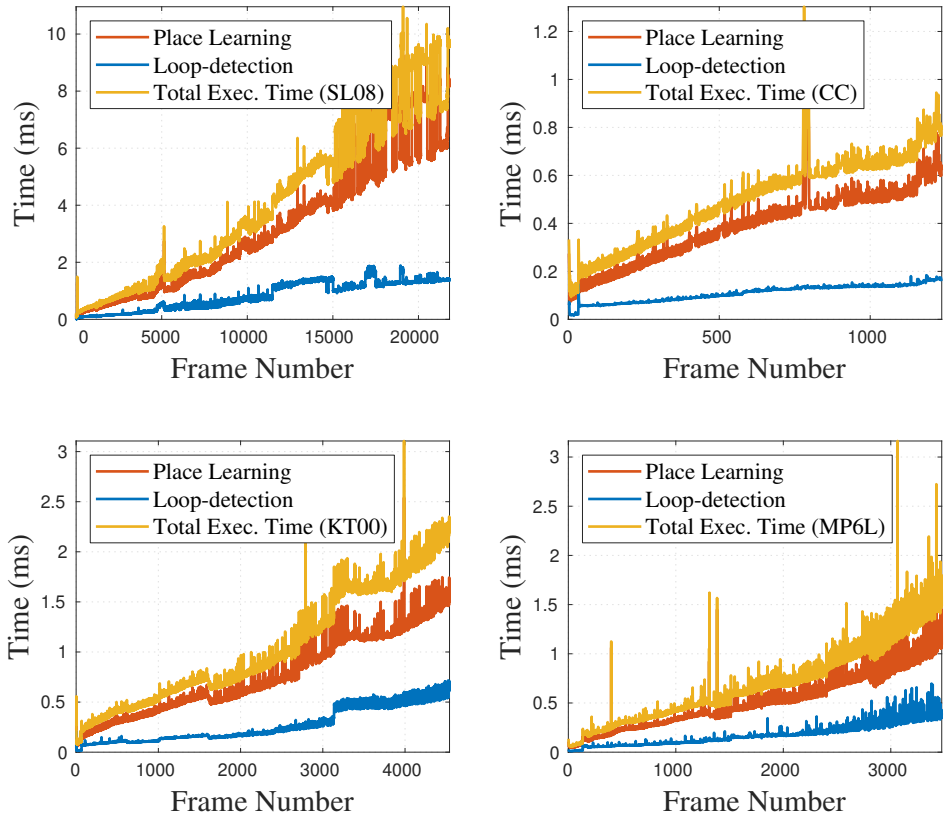


Figure 6.9: Execution time of IGBN across the selected datasets. The depicted time profiles show the execution time of the learning and recognition algorithms as a function of frame number. The sequence SL08 being the longest sequence has the average the execution time of 4.68 ms (Kazmi and Mertsching 2019a).

instance, follow the gap between the learning and the total execution times along the vertical axis. The longest sequence among the datasets is SL08, which comprises ca. 22K frames along the traversed route. At this sequence, the total execution time remains between ca. 8−10 ms near the end. On the other hand, for the CC test sequence, our algorithm spends the least execution time, i.e., ∼0.8−1 ms. Some of the timing fluctuations ascribe to the winner search, neuron creation and network pruning processes. Whereas, other fluctuations occur due to time slots scheduled for the operating system's priority processes. Although the increase in execution time across the test sequences is linear, it is alarming to the goal of long-term place recognition. As a possible remedy to this matter, one could make use of hash tables, but this is the aspect of the future research.

We now compare the agility of the baseline methods with our current and previous approaches. Table 6.4 enlists the average execution times of the algorithms on the representative test set. Note that the feature extraction time, including all the preprocessing overheads, is already included in the average execution times of our approaches. Our new approach, i.e., IGBN, performs faster than GBN on all the test sets, whereas this difference is more noticeable on the longer sequences,

Table 6.4: Average Execution Time (ms) on the Repre–sentative Datasets (Kazmi and Mertsching 2019a)

Approach	SL08	CC	KT00	MP6L
IGBN	**99.1**	95.4	96.9	95.6
GBN	113.4	96.7	99.6	97.9
iBoW-LCD	259.6	175.5	277.1	339.3
FAB-MAP 2.0	299.0	259.7	388.1	523.6
SeqSLAM+gray	172.3	**39.3**	**57.2**	**47.0**
SeqSLAM+gist	112.0	97.9	107.8	103.7
SeqSLAM+CNN	619.3	125.5	573.9	424.3

e.g., SL08. The improvements in the average execution times can be attribute to several factors. For instance, relaxing weight normalization constraint wipes off the square root computation and the division operation. Among all the operations, the most expensive part in our algorithm is the time to extract the gist features. On average, the gist feature extraction takes 94.8 ms, which includes all the image preprocessing steps, such as grayscale conversion, down-sampling, cropping and patch normalization.

On small test sets, i.e., CC, KT00 and MP6L, SeqSLAM+gray performs relatively fast. Whereas, the execution time of the SeqSLAM's variants increase on the long routes, i.e., SL08, which is more apparent for the SeqSLAM+gray. Note that the computation of the gist features takes more time compared to obtaining the gray representation of an image. On the contrary, the dimensionality of the gray features is larger than the gist descriptor. This means that the more comparison time will be needed in case of gray features (due to dimensionality). As a result of above differences, on small datasets, the gist-based search will consume more time because the algorithm spends more time computing the gist features. Hence, SeqSLAM+gray is faster than SeqSLAM+gist (and even our approach) on smaller datasets. On the contrary, for a large number of examples in the search space (database), as is the case with SL08, the vector comparison time dominates the feature extraction time due to dimensionality of the vectors. Thus, SeqSLAM+gist performs fast in this scenario relative to the gray variant. The effect of the dimensionality on execution time is further observable for CNN-based variant of SeqSLAM. The vocabulary-based methods, i.e., iBoW-LCD and FAB-MAP, show almost constant time. However, the increase in runtime for KT00 and MP6L is the consequence of high image resolution.

6.5.5 Discussion on System's Parameters

This section analyzes the performance variation of the system in relation to the critical network parameters. In this regard, finding the threshold range that delivers the best

precision–recall results across the datasets is important. To this end, we obtain the precision–recall coordinate pairs for each dataset, in accordance with the procedure described in Section 6.5.2. The generated points are segregated into non-overlapping sets of discrete intervals. For each interval, we compute the mean scores and the standard deviations. The resulting distribution of the precision and recall scores across the intervals is plotted in Figure 6.10, with the corresponding mean scores and the error bars. The first threshold interval has highest variability in precision scores, while in the second interval this reduces significantly. However, the mean precision scores never reach 1 until third interval. For the 100% precision criteria, the optimal threshold (δ) exists in the right-closed interval (0.2,0.3]. Selecting $\delta >$ 0.3 still asserts 100% precision, but then variability in recall scores increases.

For further analysis, we fix $\delta = 0.3$ and perform a systematic evaluation of the important network parameters, namely initial learning rate α_0, kernel bandwidth β, error tolerance τ and the smoothing factor γ. In this procedure, all the other parameters are set to their default values listed in Table 6.2. The precision–recall performance of β is evaluated in the range $32-D$, with the increments of 32. Regarding α_0 and τ, we vary the parameters' values in the range $(0, 1]$, considering the step size of 0.1. The evaluation interval of the parameter γ spans over the range γ_0-1, where 1 applies no smoothness and γ_0 selects the default smoothness level for the test sequences based on their frame rates, given by (6.2). Figure 6.11 depicts the performance variation of the algorithm for each parameter across the representative test set.

In general, high precision scores persist for the parameters α_0 and β, compared to τ and γ, irrespective of the test sequence. At a low learning rate, the network instantiates neurons more frequently to adapt itself according to the distribution of

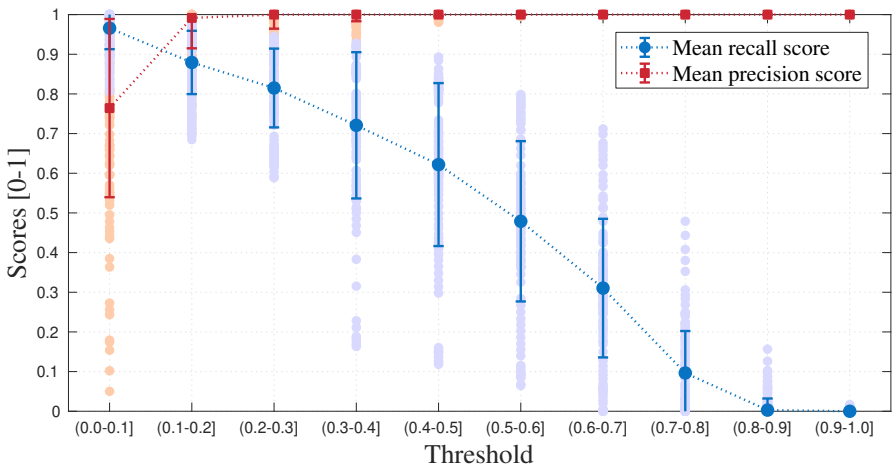


Figure 6.10: Change in precision–recall performance of the algorithm in relation to threshold (δ). For 100% precision, the minimum acceptable threshold lies in the range $(0.2, 0.3]$ (Kazmi and Mertsching 2019a).

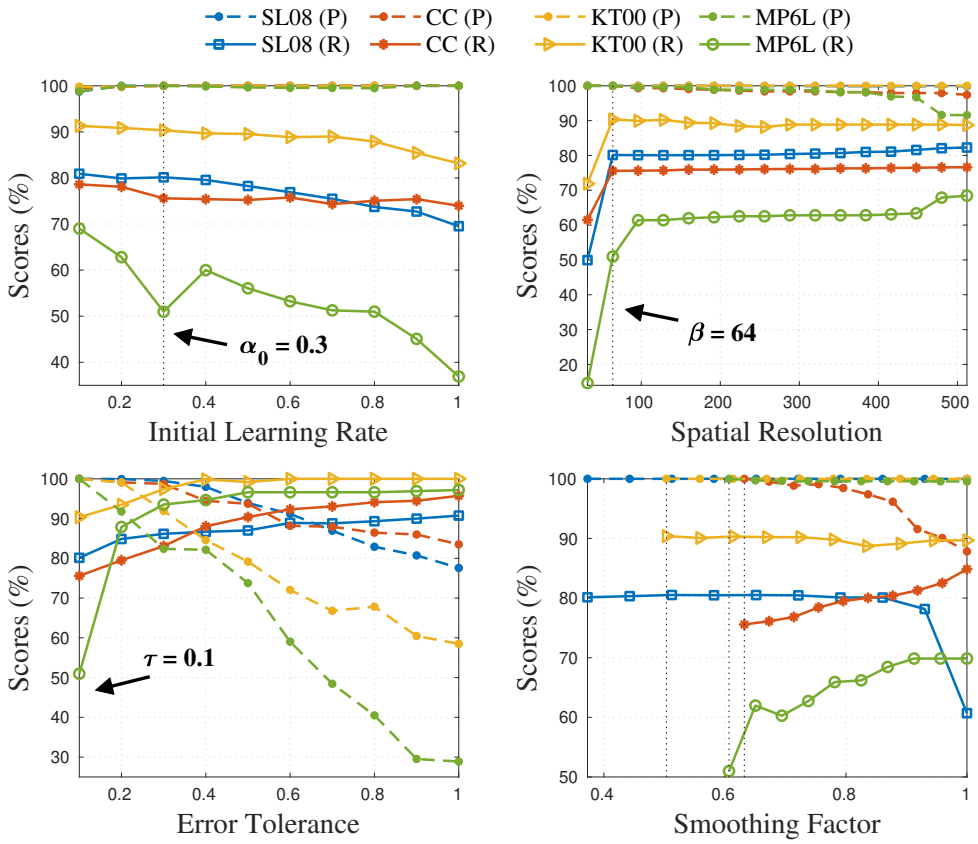


Figure 6.11: Evaluation of critical network parameters. Note the scales of the axes in each subplot. The precision (P) and recall (R) curves in the subplots are obtained by varying the corresponding parameters (see text), across the datasets. The precision curves are shown as dashed lines, while the recalls as the solid lines (Kazmi and Mertsching 2019a).

data points in the feature space. Here, the frequency sensitive learning scheme, i.e., Eq. (6.7), offers a notable advantage. For example, the learning rate of a neuron is decaying function of its learning frequency. Hence, if a neuron initially has a low learning rate, its learning rate will rapidly drop off, subject to its subsequent selections as a winner. One can observe in the plot that for $\alpha_0 < 0.3$, the precision scores are slightly less than 1. However, large initial values of the learning rate (i.e. > 0.5) degenerate the recall scores. This happens because large α_0 values are more likely to cause an overshooting problem in the gradient descend. That is, the neurons are dragged farther away from the input. One can assess this behavior in Eq- (6.6) by setting the initial learning rate to larger values. The vertical dotted line in the figure refers to the cut off bar for the initial learning rate. Note that at this position, the precision–recall rates across the datasets are consistent with results reported in Table 6.3.

The parameter β configures the bandwidth of the kernel in Eq. (6.6). In Figure 5.4, we elaborate the effect of kernel bandwidth on topological connections between the neurons. Here, we discuss how the β values influences the overall system's preci–

sion–recall performance. Increasing the kernel's bandwidth beyond the threshold bar gradually deteriorates the system's precision across the test set. In particular, this behavior is more notable for the CC and MP6L sequences. By contrast, the sequences with high frame rates, i.e., SL08 and KT00, still show 100% precision. Moreover, one might have noticed that, for $\beta > 64$, recall scores do not improve notably for any test sequence. Generally, we say that recalls degrade for $\alpha_0 > 0.3$ or $\beta < 64$. Indeed, the kernel bandwidth also has an effect on the likelihood, given by (6.13). Hence, as $\beta \to D$, the likelihood that a particular data point is representable by a neuron becomes nearly zeros. On the basis of experimental evaluations, we conclude that one should avoid high β values (esp. $> D/4$), in particular for the sequences with low frame rates. Such a choice ensures that recalls do not degrade and the sum log-likelihood does not become zero.

For error tolerance τ parameter, we observe a rapid decline in the precision levels across the datasets, compared to other parameters. Large τ values indicate that a neuron will accept more data points for learning, which in other words expands the quantization span of a neuron in the feature space. As a result, less neuron will be instantiated to represent the data. The end-result is that the learned representation are coarser and under-represent the topological structures of the data points in the feature space. This is the reason why precision tappers off rapidly for $\tau > 0.1$.

With regard to the smoothness factor γ, the dotted vertical bars in the plot refer to the default smoothness levels of the corresponding datasets. The figure shows that the recall scores are highly variable between the sequences. For instance, increasing γ beyond the default smoothness level improves the recall in some cases (e.g. CC and MP6L), whereas for the sequences with high frame rates (i.e. SL08 and KT00), the recalls degrade. By contrast, the precision levels across the representative sequences remain slightly less than 1, as the smoothness factor exceeds the default level. A curious reader might have noticed that $\gamma = 1$ is a no smoothness condition. In this case, our algorithm uses the gist descriptor of only the current location to represent a scene. Thus, the nearby context is not incorporated in the process of capturing the spatial layout of the scenes. The results in the figure, hence demonstrate the significance of the nearby context for loop detection.

6.5.6 On the Effect of Weight Normalization

In Figure 6.3 and the corresponding paragraph, we state the technical reasons for not normalizing the weight vectors. This section presents the quantitative analysis of algorithm's performance regarding the normalization and no normalization cases. In evaluation, we also consider our previous approach, i.e., GBN (Kazmi and Mertsching 2016b). For a comparison between normalization and no normalization scenarios,

Table 6.5: Effect of Normalizing Network Weights (Kazmi and Mertsching 2019a)

Criteria	Datasets	With Normalization		Without Normalization	
		IGBN	GBN	IGBN	GBN
Network Size (N)	SL08	5041	5807	**2787**	3840
	CC	603	726	**434**	443
	KT00	1649	1814	**1005**	1309
	MP6L	1045	1069	**700**	769
Avg. Recall at 100% Precision (%)	SL08	24.51	1.82	**44.47**	2.43
	CC	27.70	–	**39.33**	–
	KT00	29.08	–	**56.34**	3.31
	MP6L	19.04	–	**21.0**	2.80

– Average recall cannot be calculated, as 100% precision is never reached.
∗∗ Note that weight normalization results in more neurons because the squared error metric is no more useful in a normalized space. One should rather use a different similarity metric. Besides this, the normalization tends to complicate the clustering of the data points (see Figure 6.3).

the network size and the average recall scored by the algorithms at 100% precision are observed; where network size is the number of neurons in the network. The results obtained for the representative sequences are summarized in Table 6.5.

A common behavior between the algorithms is that weight normalization produces more neurons. Hence, the network size is increased. By contrast, leaving the weights unnormalized results in less number of neurons for representing the feature space. Note that SL08 sequence has 21,815 frames, but IGBN instantiates only 2787 neurons for the no normalization scenario. Whereas, the normalization creates 5041 neurons to learn the same set of frames. Considering the no normalization case, it is worth mentioning that our new approach, i.e., IGBN, utilizes 2787 neurons compared to 3840 neurons created by GBN (our previous work). Achieving such a compression of the feature space is partially attributed to the smoothing factor, as high smoothness levels are related to the creation of less neurons.

An important question that needs further consideration in relation to above dis–cussion is the impact of compression on the classification performance. For this purpose, we record the average recalls of the algorithms at 100% precision. The advantage of this evaluation measure is that it includes only those recall scores in the average for which no false detections exist. One can follow in the table that without normalization the average recall rate considerably increases for all the test sequences. Although for GBN this improvement is comparatively small, one can

track down the examples in the no normalization column where GBN has started to shows 100% precision. Hence, average recall has slightly improved.

6.5.7 Correctness of the Learned Representation

In this section, we evaluate the progression of error in the network for each input. This analysis is of particular importance for IGBN, as we have improvised the compression ratio of the feature space. Thus, the reduced error would validate the correctness of the learned representation. As a reference, we include the comparison with our previous algorithm (i.e. GBN). The network error in the algorithms is computed using Eq. (5.13). Figure 6.12 depicts the error profiles of the algorithms as a function of frame number. Notice that the IGBN network always starts in a large

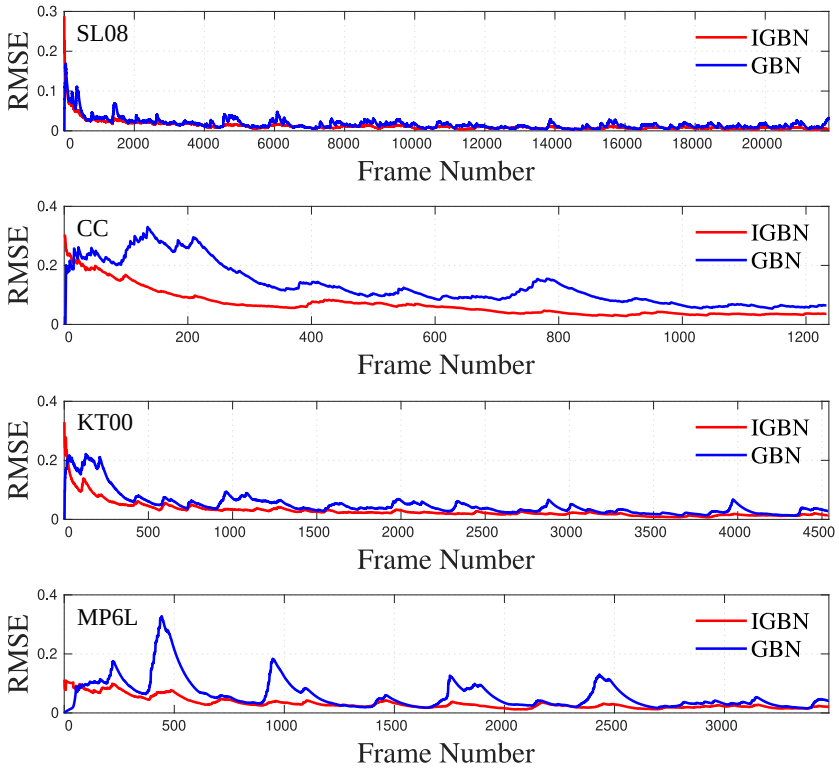


Figure 6.12: IGBN vs. GBN: Error regulation in the networks as a function of frame. Our improved version (i.e. IGBN) consistently shows less error across the datasets than our previous algorithm, namely GBN.

error across the datasets compared to GBN. It happens because the first descriptor of the neuron is sampled according to Eq. (6.1), where the sampling factor is set automatically subject to the frame rate of the sequence. In comparison, GBN uses a hard-coded value to initialize the neurons. Later, both the algorithms have managed to regulate the error to a lower amount, but IGBN persistently maintains lower error levels. Regarding GBN algorithm, the severe error fluctuations are observable for the CC and MP6L sequences, which are recorded at relatively lower frame rate. The

exponential smoothing and the new scheme to instantiate neurons in the feature space assist IGBN to keep low error profiles even at lower frame rates.

6.5.8 Evolution of Neurons in the Network

Besides the network dynamics, there are multiple external factors due to randomness in environments that significantly influence the network's growth. To this end, we evaluate the growth of the neurons in network as a function of the image frames, as plotted in Figure 6.13. A noticeable common aspect among the test sequences is that the network's growth follow almost a linear trend. The main diagonal in the figure represents the recently created neurons that are selected as a winner to learn from the input. By contrast, the progressively aligned neurons below the main diagonal refer to the winner neurons that are created in past, but they are selected again to represent the current frame. Many similar constellations in the plot reflect the persistent utilization of the network, and thus validate the correctness of the learned feature maps.

Here, we would like to clarify that the winner neuron does not necessarily reflect a revisit to a familiar location. For instance, follow several isolated points below the main diagonal. Alternatively, notice the horizontal sequence of points in the zoomed windows in the subplots of each test sequence. These are those cases in which the recently created neuron is continually selected as a winner over the successive frames. We already clarify this concept in the previous chapter, follow details in Section 6.13. Here, we briefly describe this difference for completeness reasons. Note that winner neuron is obtained using Eq. (6.4), whereas for place recognition we use the activation strength of the maximizer neurons, which is obtained from Eq. (6.12). On that ground, one should be careful between the winner and maximizer (i.e. highly active) neurons. They conceptually refer to different aspect of the system.

Inside the zoomed regions, one may have noticed few neurons that are recently created along the revisited routes. Such a response of the algorithm reflects back to uncontrollable changes in environments, such as appearance variation originating from shadows, poor lighting conditions, dynamic objects or large viewpoint changes. Among the representative sequences, SL08 is the longest one with many loop closing routes, and several of them are traversed multiple times. Thus, it demonstrates the maximum utilization of the network. In general, our algorithm shows the highest compression ratio on SL08 sequence. The sequence comprises 21,815 frames, while 2787 neurons are created to learn the environment. This means that on average a single neuron learns 7 to 8 frames. On the other hand, image-to-neuron compression ratio for CC is 1237: 434. Thus, on average $2-3$ frames are assigned to a neuron. At first sight, this concludes that more compression is selected for the sequence with

higher frame rates. Although this is partially true, one cannot still generalize this behavior. Indeed, the creation of neurons also relates to many other factors, such as number of traversal through a familiar route, scale of the environment, lighting situations, dynamics objects and vehicle's velocity. For instance, consider the KT00 and MP6L sequences, where the former has a higher frame rate than latter. Despite such a difference, the average frame assignments per neuron for these sequences range over $4-5$ frames, i.e., $4541:1005$ and $3474:700$, respectively. It happens because higher frame rate to some extent counterbalances the velocity gains.

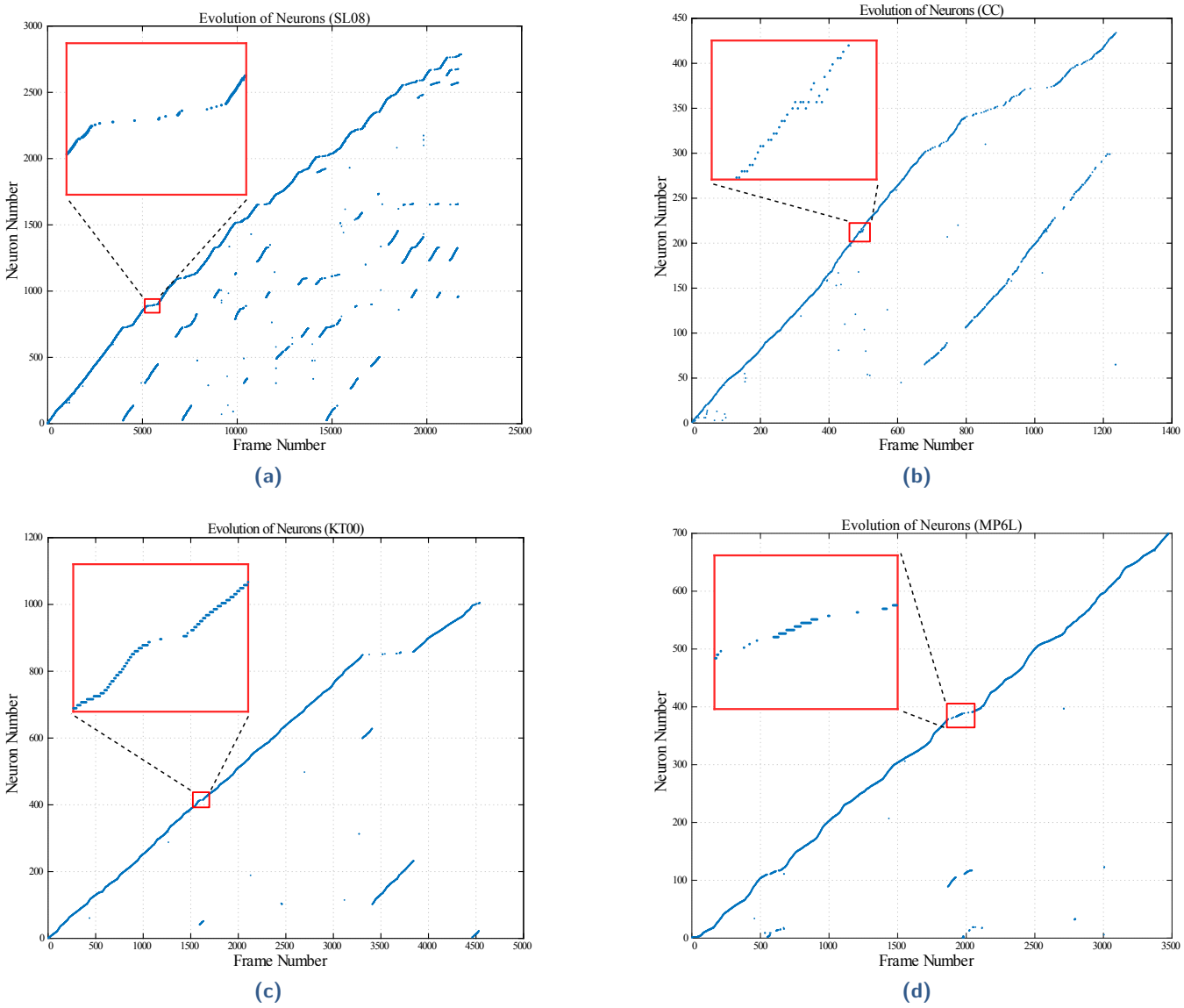


Figure 6.13: Evolution of neurons across the selected sequences. The selection of the winner neuron is plotted as a function of frame number (Kazmi and Mertsching 2019a).

6.5.9 On Extensibility to Other Feature Spaces

This section investigates the possibility of extending the IGBN to heterogeneous feature spaces (e.g. HOG features), without following an explicit parameter tuning procedure. The steps for extracting the HOG features are already explained in Section 2.6.4. We analyze the performance of the gist- and HOG-based variants of IGBN, denoted as IGBN+gist and IGBN+HOG, respectively, on the representative test sequences. Throughout the experiments, all parameter settings are held fixed. The maximum recalls achieved by the algorithms at 100% are depicted as a bar plot in Figure 6.14.

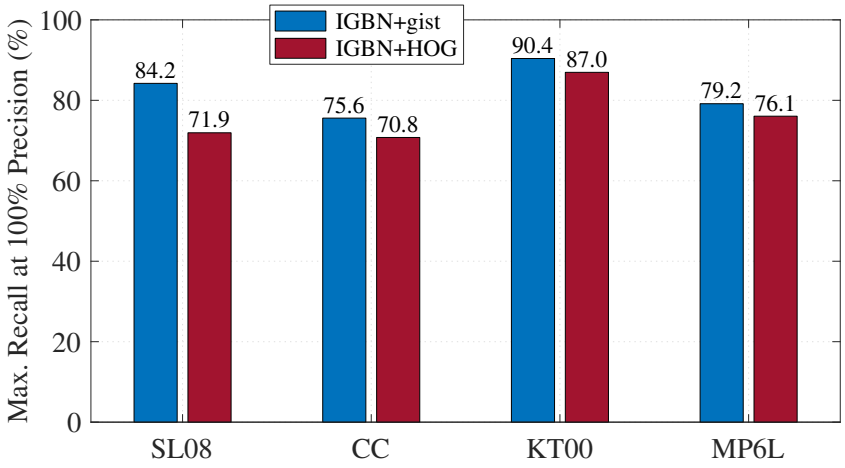


Figure 6.14: Maximum recalls achieved by the algorithms at 100% precision. The plotted scores are rounded off to the first decimal digit (Kazmi et al. 2019b).

Although the HOG-based variant of IGBN does not perform better than its gist-based version, the results across the datasets are still quite comparable to IGBN+gist. Here, we would emphasize that the HOG descriptor is only 192-dimensional, compared to a 512-dimensional gist vector. Despite such a low dimensionality, the HOG-based visual description maintains sufficiently high recalls at 100% precision. We partially attribute this robustness to the exponential smoothing. This demonstrates the algorithm's strength to learn the topological structure of the new feature spaces, without the overhead of tuning the network parameters. Note that it is a direct transfer of the network to the feature space does not payoff. In order to skip the parameter tuning and translate the system to a new feature space, one has to at least consider the scaling factor across the dimensions of the descriptor, such that the numerical values in each dimension of the descriptor lie between 0 and 1.

An obvious question that may arise here is that does increasing the dimensionality of the HOG descriptor help? In our experiments, we noticed that a very high-dimen–sional HOG descriptor does not enhance the system's place recognition performance. Rather a high-dimensional descriptor constitutes very localized gradient details in

images. The resulting descriptor is therefore sensitive to tiny variations in images caused by lighting situations or moving objects. On the other hand, a low-dimen–sional description coarsely samples the gradient information from the images. Hence, care must be taken as a very low dimensionality is insufficient to uniquely describe the scenes, and may lead to performance degradation.

Another important discussion element is that of the memory requirements and timing efficiency of the two systems, i.e., IGBN+gist and IGBN+HOG. In that capacity, we record size of the network in physical memory and the mean execution times of the algorithms. The results are listed in the Table 6.6.

Table 6.6: IGBN+gist vs. IGBN+HOG: Space and Time Requirements (Kazmi et al. 2019b)

Dataset	Network Size‡		Mean Exec. Time¶	
	IGBN+gist	IGBN+HOG	IGBN+gist	IGBN+HOG
SL08	5.44	6.01	99.1	33.45
CC	0.85	0.71	95.4	29.91
KT00	1.96	1.74	96.9	30.15
MP6L	2.73	1.4	95.6	30.0

‡ Size of the network in physical memory in megabytes (MB)
¶ Execution time is measured in milliseconds (ms). Due to low-dimen–sionality of the HOG features, our algorithm spends less time in feature extraction, learning (incl. winner search) and place recognition. Thus, execution times are less affected by small variations in the network size.

On majority of the test sequences, the IGBN+HOG reserves relatively less physical memory compared to its gist-based counterpart. However, on longer routes, i.e., SL08, the representation learned by HOG-based variant occupies more memory (i.e., approx. 6 MB). In fact, the increased network size here is linked to the creation of more neurons in case of IGBN+HOG. Regarding the algorithms' agility, clearly IGBN+HOG is far more efficient than IGBN+gist on all the test sequences. Note that the mean execution times of the algorithms, listed in the table, already include the feature extraction time (and the time for preprocessing overheads). Even on the longest sequence (i.e. SL08), the execution time of IGBN+HOG is just 33.45 ms compared to IGBN+gist. In general, we say that HOG-based variant of IGBN is 3 times faster than the gist-based variant. Hence, we achieve three-fold speed-ups in execution time using HOG features, while still retaining almost 93% of the average place recognition performance relative to gist-based variant.

6.6 Summary

This chapter deals with the challenges of online visual learning and recognition in unknown environments. In this respect, we proposed significant improvements in our previous approach, i.e. growing belief network (GBN), and thus named the improved version as IGBN. Rather than capturing the spatial layout of a location from a single frame, exponential smoothing is applied. Therefore, in the new algorithm, the layout of a location is obtained as a weight average of the gist-based description of the nearby locations. The new learning dynamics of the GSOM facilitates the network to evolve as new data appears. Consequently, the network has been able to learn a compact representation of the input space without degrading the recognition performance. The IGBN algorithm differs from GBN with regard to the network initialization strategy, learning scheme, weight update rule, and the novel approach to instantiate the new neurons.

Furthermore, for robust place recognition, we now look out for a sequence of minimizer neurons and utilized their activation strength to compute the expected network activity; where the activation strength of a single maximizer neuron is calculated from a well-grounded Bayesian formulation. Finally, a query place is determined as a loop or no-loop location, if the expected activity of the network satisfies the predefined threshold criterion.

A sequence of experiments are performed on challenging dataset of varying level of complexity. The experimental results demonstrate the general applicability of the IGBN across the environments, without following any environment-specific parameter tuning or offline training procedures. On the longest routes (as large as $\sim$18 km), our algorithm has demonstrated its real-time performance and achieved top scores compared to the baseline methods. Further experiments showed that IGBN can also be fused with other feature spaces (without parameter tuning), provided one takes into consideration the scaling factor along the dimensions of the descriptor. This signifies the feature-agnostic characteristic of our new algorithm.

7 Conclusion and Future Research

This section summarizes the contributions of this research in the order of progressive improvements that are demonstrated in Chapters 4, 5 and 6, respectively. As this research is partly inspired by biology for scene learning and recognition, in that capacity we discuss the plausibility of the biological processes in a technical system. Thereon, we tabulate the novelties of our work in relation to the state-of-the-art methods and discuss the prospects of further developments to deliver an even robust and long-term solution to place recognition.

7.1 Thesis Summary

Correct recognition of familiar places, aka loop-closure detection, is by far the biggest challenge to long-term navigation. It is of particular importance for robotic mapping, as a robust loop detection system ensures the global consistency of the map. The complexity of the matter arise from the fact that the environmental conditions are unpredictable and dynamic. For instance, a place may look different at different daytimes due to lighting variations, moving objects or appearance changes. On the other hand, the more we explore the environments, the more clumsy and perceptually similar they look.

Conventionally, the loop closing task has been addressed in the coordinate space of the map. However, the recent research has shown that the loop detection in the space visual appearance is far more robust and efficient than the loop detection in the geometric space, follow details in Section 2.5.3. This is where the roboticists are divided between producing the geometrically accurate map or learning topologically correct maps in the space of visual appearance, aka appearance-based mapping. In principle, appearance-based methods are analogous to how biological systems, e.g., humans, recognize places and navigate through diverse environments without the notion of precise geometry.

As this research is primarily inspired by biology, we follow on the appearance-based mapping. Hence, in all of our contributions, we do not utilize or produce the metric data (e.g., GPS, visual odometry, etc.). The problem that continue to exist across many state-of-the-art methods is their dependability on the environments. In fact

this issue emanates from the requirement of environment-related parameter tuning or offline training phase prior to the system's deployment. The consequence of these extensive procedures is that the existing systems cannot readily be transferred to the previously unobserved environments. Even if, these systems are deployed, they suffer severe performance degradation.

In this research, all of our contributions are aimed at dealing with the challenges of online visual learning and recognition in unknown environments. In this respect, our first contribution was an online bio-inspired place recognition front-end for the RatSLAM system. The RatSLAM algorithm models the biological processes of a rat's brain that underpin the navigation. We found out that the place recognition module of the RatSLAM is not biologically inspired. Moreover, it lacks the aspects to learn from the visual input. Rather the cells, which have authors have named the local view cells, are the visual templates that are generated based on ad-hoc threshold. Taking the inspiration from biology, we introduced a technical system that incremen–tally learns from the input and performs on the fly place recognition in unknown environments. Instead of following the fine-grained image analysis or time-con–suming segmentation process, the human-like perceptual layout of the scenes was extracted from images using gist features. For learning the scene representations, a modified growing self-organizing map (GSOM) was proposed – a type of neural network that emulates the competitive behavior of cells found in cortical regions of the brain. Finally, the proposed GSOM-based front-end (GSF) was integrated as a data association module to the RatSLAM system. The results of the experiments demonstrated that the proposed algorithm generally achieved top scores, showed loop detection latency of less than 1.6 seconds, and on the average instantiated 20% less view cells compared to the baseline place recognition system.

Next, we introduced the concept of simultaneous visual learning and place recogni–tion for appearance-based mapping. In this regard, several major changes were made in the learning dynamics of the GSF. The place recognition was now handled by a novel probabilistic framework, which relies on maximum a posteriori estimate (MAP) instead of naive nearest neighbor search (NNS). The resulting system was altogether referred to as a growing belief network (GBN). Learning dynamics of the GBN differ from GSF in several ways, such as winner-takes-all mechanism was replaced by the winner-takes-most learning scheme. This was achieved by allowing the interaction among neurons. The spread of the network was controlled through the error rather than ad-hoc threshold criterion. Inactive neurons were periodically pruned from the network to free the occupied resources. The performance of the system was evaluated on seven challenging sequences of varying complexity. The experimental results demonstrated the strength of the GBN to perform real-time place recognition in unknown environments, without environment-specific parameter adjustment. Moreover, the place recognition performance of GBN has generally outperformed the

state-of-the-art approaches in F_1-measure. During the $\sim$25 minutes drive through the suburb of St. Lucia, on average GBN created 73% less neurons than the number of frames generated in that time. This signified the GBN's ability to learn the compact representation of the large-scale environments.

Thereon, we extended the existing learning and recognition algorithms with the concept of nearby context. The proposed improvements significantly enhanced the GBN's performance and further cut down the memory requirements. Henceforth, the improved version of GBN was referred to as IGBN. The new approach differs from the previous one in the following aspects. We complemented the spatial layout of a location with the visual appearance of the nearby places. For the decision of a loop detection, we aggregated the activation strength of a series of maximizer neurons, which was termed as expected network activity. Such a scheme is more reliable than observing the response of only a single maximizer neuron. We also proposed the frequency sensitive learning scheme so that the neurons which have learned the large portion of data do not change their position in the space. This is important, as otherwise the spurious inputs or outliers distort the learned map and thus lead to a large quantization error. To instantiate a new neuron, the learned network statistics were used, which has significantly reduced the error progression in the network. Lastly, unlike GBN, the weight vectors were left unnormalized. This has improved the space and time requirements of the IGBN, and at the same time enhanced the place recognition results in terms of average recall at 100% precision.

To demonstrate the significance of the proposed changes, an exhaustive set of ex–periments were performed on eleven test sequences that involve various challenges, such as extreme brightness changes, moving objects and perceptual ambiguities. In all experiments, the parameter settings were held fixed. The results signified across the environments applicability of the proposed algorithm, without environment-re–lated parameter tuning or offline training. On the routes, as large as 18 km, the algorithm showed real-time performance and achieved top scores compared to GBN and the popular state-of-the-art methods. When tested in the new feature space, the algorithm delivered the feature-agnostic performance. This is because in the new space the algorithm's place recognition results are roughly on the par with the gist space, while gaining threefold speed-ups in execution times. Table 7.1 enlists the technical improvements that are successively applied to the proposed algorithms.

7.2 Biological Fidelity in a Technical System

At this point in discussion, one might stop and bring into question the completeness of the present biological findings, and if the existing research practices have achieved

Table 7.1: Summary of Improvements in the Proposed Algorithms

Aspect	GSF	GBN	IGBN
Scene description	Single frame-based	Single frame-based	Nearby context
Start-up neurons	Single	Multiple	Single
Initial weights	Descriptor's replica	Ad-hoc sampling	Automatic sampling
Learning rule	Winner-takes-all	Winner-takes-most	Winner-takes-most
Learning decay	Neighborhood aware	No decay	Frequency aware
Neural connectivity	Distance-based	Gaussian RBF	Gaussian RBF
Growth control	Ad-hoc	Error-based	Error-based
Neuron instantiation	Ad-hoc	Ad-hoc	Error driven
Network pruning	✘	✔	✔
Normalizing Weights	✔	✔	✘
Recognition	NNS	MAP	Expected MAP
Mapping Domain	Weakly-metric	Appearance-based	Appearance-based

The learning decay in GSF is regulated by the neighborhood size of the winner, whereas for IGBN the learning decay is inspired by frequency sensitive learning. For GBN, the learning rate was kept constant.

enough to emulate the biological processes, underpinning the navigation, in a technical system. In the beginning of our discussion, see Section 2.7, we learned that the brain's regions such as occipital and temporal lobes are central to the tasks of visual perception, learning and navigation. The fact that these regions are essential to the navigation tasks does not imply by any means that the neural connectivity and behavior in these regions is localized to them. These regions work in coordination with other areas of the brain and the modern-day neuroscience has to discover yet the working principals in which they coordinate.

Besides this, the visual perception theories are mainly based on the experimental studies. There exist two competing theories regarding the process in which visual information is perceived by brain, such a bottom-up and top-down processing. Although the researchers agree on the formation of gist of a scene in iconic memory, they disagree on how this information is later used for memory formation. For instance, the gist of the scene is computed along the ventral streams, whereas the saliency maps are computed along the dorsal pathways. It is now important to explore how the feedback from saliency maps influences the transfer of gist further down the temporal lobe. Otherwise, one can argue if the gist formation is a temporary process and does not further help the visual learning. This suggests, although the current biological findings have evolved to the betterment, several questions remain unanswered and need further study.

Unfortunately, whatever is known about the biology of visual perception and navigation cannot be efficiently transferred to the technical systems. For instance, human brain is a complex machinery with trillions of neurons. Take an example of self-organizing map (SOM), which models the behavior of certain cells in the visual cortex

and the areas of hippocampus. Realizing such a neural network to map large-scale environments demands extensive computational resources. Hence, we opted for the GSOM, and so in that sense the developed system can be loosely considered as bio-inspired but not strictly biological. In addition, in biological systems, the acts of memorizing and recalling places happen subconsciously. However, teaching these skills to a technical system needs serious efforts. For instance, there is no general definition of a place. A place may be interpreted differently at different levels of abstraction. At a country level, several cities are considered as places, while inside a city famous landmarks could be the individual places, and this varies further down the hierarchy.

7.3 This Research vs. State-of-the-Art

To conclude the standing of this research in relation to the state-of-the-art, the St. Lucia suburb (SL08) is chosen. Indeed, it is the longest traveled route (i.e. $\sim$18 km) in the benchmark datasets, has numerous loop locations and many of them are vis–ited multiple times. Thus, this sequence serves as a good starter to summarize the performance of the proposed and baseline algorithms for large-scale visual learning and place recognition. Table 7.2 sums up the essence of the information provided in Tables 3.1, 6.3, and 6.4. For further details, we refer to the respective tables.

Table 7.2: Summary of Comparison: Proposed vs. Popular Appearance-based Methods

Aspect	**IGBN**	**iBoW-LCD**	**FAB-MAP**	**SeqSLAM**
Features	Gist	ORB	SURF	Gray
Learning Method	GSOM	Hierarch. Trees	K-means; Chow Liu	–
Training Proc.	Real-time	Online	Offline	–
Recognition	Exp. MAP	NNS	Bayes Filter	DTW
Avg. Runtime (ms)	**99.1**	259.6	299.0	172.3
Max. Recall (%)	**80.13**	41.32	53.98	67.96

The table includes only the comparison with the SeqSLAM+gray, as it achieves the top scores among other variants of SeqSLAM on SL08, see Table 6.3. The maximum recalls reported in the table are consistent with 100% precision criteria. The dash symbol (–) indicates that this approach does not involve learning. It is basically a sequence matching method.

The approaches that use global descriptors, i.e., IGBN and SeqSLAM, in general achieve better place recognition performance than methods that use local features, namely iBoW-LCD and FAB-MAP. Note that iBoW-LCD, which learns in an online fashion, has been a low performant method among all in the St. Lucia suburb, see Table 6.3. The reason is that iBoW-LCD is grounded in binary features, whereas binary features are sensitive to brightness changes. On the other hand, SeqSLAM does not involve learning. Although it handles with the sequences in an incremental manner, but its timing efficiency drops rapidly as the size of the database images gets

large. For instance, on small-scale datasets, see Table 6.4, the gray-image variant of SeqSLAM has quiet fast query processing time. Nonetheless, shifting the system to a large-scale environment has reduced the efficiency almost by three times. By contrast, our approach, i.e., IGBN, demonstrates high efficiency than the baseline methods even on the large datasets.

7.4 Future Research Directions

This sections discusses the possible ways in which the present research can be further extended.

7.4.1 Search Optimization

In Sections 4.4.3, 5.4.3, and 6.5.8, we concluded that the growth of the network is not only related to the scale of the environment, rather it is also modulated by other influences, e.g., lighting condition, dynamic objects, speed of the vehicle and frame rate. Hence, the network's behavior along the routes is non-deterministic. An optimistic side of the current network dynamics is that the network's growth is not exponential. From the perspective of a long-term place recognition, the ever-increasing size of the network demands fast or even clever search algorithms so that the the winner neurons can be efficiently retrieved from the network. A direct application of the approximate neatest neighbor search (ANNS), or other advanced techniques, e.g., locality-sensitive hashing (LSH), would not help. The reason is that the neurons are under constant flux of adjusting their position as the new data points appear. Hence, to make these optimization techniques effective, careful procedures need to be defined.

7.4.2 Extension of the Place Recognition Subsystem

At present, the probability distribution in Eq. (6.11) is uniform. Practically, one can model this probability distribution with a transition table $\mathbf{T}$, in which each entry t_{ij} denotes the probability of transition from a neuron i to an arbitrary neuron j. Thus, given the activation strength of the two maximizer neurons at time instants $t-1$ and t, the table would determine how likely it is to move between these neuron; where the transition probabilities between neurons can either be modeled with the Gaussian RBF, given by Eq. (5.7), and further supported by the temporal information, such as how closely they were instantiated in time. This leads to fusing the hidden markov model (HMM) in the current place recognition scheme.

7.4.3 On Learning the Geometric Maps

In the present setup we addressed the challenges of online place recognition for pure appearance-based mapping. However, there exist some hybrids, e.g., (Pepperell et al. 2016), that utilize the visual odometry to improve the place recognition results. On the other hand, others, e.g., (Jacobson et al. 2015), use the place recognition system as a loop detection module to obtain the geometrically consistent maps.

In this context, our work can be further integrated as a front-end to the pose-graph optimization techniques, if the metric mapping is desirable. On the opposite, the visual odometry information can be added to the transition between the neurons for improving the place recognition results. This, for instance, can be achieved by retrieving the local features from the images that are mapped to the consecutively active maximizer neurons. Thereon, on can build the co-visibility graph of features that are common between the frames. Finally, the geometric validity constraints be–tween the images can be enforced. This is expected to cut down the false detections, while increasing the accuracy of the place associations. Also, using the local features would reinforce the algorithm's ability to handle large viewpoint changes.

7.4.4 Optimality of the Hyperparameters

The analysis presented in Section 6.5.9 concluded that our present approach, i.e., IGBN, is feature-agnostic. Although the results summarized in Table 6.6 point to threefold improvements in timing efficiency of the algorithm in case of HOG features, the memory consumption tells quite a different story (particularly for SL08). In fact, as the size of the environment increases, the HOG-based IGBN consumes more memory than the gist-based version. This reflects back to the growth of more neurons in case of HOG features.

Hence comes the question, are the parameter settings of the learning algorithm strongly related to the feature space?. As, otherwise, HOG might have even outper–formed the gist in terms of place recognition results. Here, the curious reader would realize that the learning dynamics of the algorithm are deep-rooted in the network parameters, such as error tolerance and kernel bandwidth. This leads to investigating the parameter settings for a different feature space, which is fundamentally the hyperparameter optimization problem in machine learning.

From now onwards we will call the parameters of the network as hyperparameters. In optimization algorithms, the weights of the neural network are treated as parameters, whereas the parameters that are configured before the learning starts are termed as hyperparameters. So, the goal is to optimize the hyperparameters of the network that

strongly relate to the feature space. Note that feature space centered hyperparameter tuning does not make the algorithm environment dependent. Rather the distribution of the data is interpreted differently in various feature spaces due to the scale of the dimensions and the number of dimensions pertinent to the feature space. This is the reason why, by just scaling the HOG descriptor, the HOG-based version of our algorithm has not underwent catastrophic performance degradation.

7.4.5 Then GSOM Meets the CNNs

The dynamic nature of the environments challenges the place recognition at different stages. For instance, other than momentary changes in appearance of a place, such as moving objects or brightness variations, a place may undergo severe appearance changes due to foliage or bad weather. Such changes in appearance stretch longer in time than those, which were dealt in the scope of this research. This brings in across the season place recognition challenge.

There are basically two ways in which further developments can follow. One possi–bility is to leverage the pre-trained state-of-the-art heavy deep-learning machines as a feature extractor for the GSOM. However, this development goes slightly into the direction of hyperparameter optimization. In this case, the very high-dimensional CNN features would serve as a condition-invariant image representation. However, an appropriate approach could be the back-to-back training of the GSOM. In this case, the network will learn the codevectors at specific layers, e.g., Conv3 or Conv5. Although, the network has to be trained in an offline manner, the learned represen–tations can be used as a vector of condition-invariant features observed in an image, somewhat similar to the vocabulary-based learning. For vocabulary preparation, usually k-means clustering is applied, which is a hard-assignment of data points to the clusters. Moreover, one has to select the number of clusters in k-means. By contrast, GSOM is a soft-competitive learning approach and one does not specify in advance the maximum number of neurons.

7.4.6 Other Considerations

Another solution, regarding condition-invariant place recognition, could be pre-–training the autoencoders or ICA on a large database to obtain a generic condition-in–variant model of the environments. The preliminary work in this direction includes (Biswas et al. 2019). The pre-trained representations can then be used to obtain the condition-invariant description of the scenes. Thereon, the IGBN could be trained online to learn these representation for real-time place recognition. This task partly

relies on hyperparameter optimization, as one needs to adjust the hyperparameters of GSOM according to the aforementioned feature spaces.

In the present setup, the network's growth is unbounded. Hence, it is possible to use the merging techniques so that the neurons which have fixed their positions and lie within a certain radius can be merged into a single neuron. This would help reducing neurons in very dense regions of the feature space. Also, this is the effective utilization of the resources.

7.5 Concluding Remarks

This research introduced real-time visual learning and place recognition algorithms, followed by a novel scheme for obtaining the visual description of the scenes. The algorithms are partly inspired by biology. Altogether the system is named as improved growing belief network (IGBN). Online place learning and recognition is a stochastic challenge. Due to the nature of the problem, the closed form solution is infeasible. In an unsupervised scenario, the best one can do is to account for errors in the network as the new data arrives, and thereon spread the network to learn the unexplored structures in the feature space. In this thesis, IGBN was able to handle 21,815 images in real-time, which is roughly the 18 km long journey in suburb of St. Lucia, Brisbane, Australia. The system instantiated just 2787 neurons to learn the test environment. Hence, the introduced solution is lightweight, i.e., it does not demand ample hardware resources, e.g., memory and processing power. Moreover, compared to state-of-the-art techniques, we generally achieved high-precision and faster performance.

Handling thousands of locations at once is computationally intensive. Instead of searching through entire database of places (in a conventional sense), this research introduced a hierarchical solution to tackle the place recognition problem in real-time. For instance, given the query image, finding a highly active neuron in the network is a multi-class classification challenge. Once the highly active neurons are known, the loop-closure decision is treated as a binary classification task. Finally, given that the current place is a loop location, the query place can be associated to the nearest place among the places that are mapped to the highly active neurons.

Bibliography

Alahakoon, D., Halgamuge, S., and Srinivasan, B. (2000). “Dynamic Self-organizing Maps with Controlled Growth for Knowledge Discovery”. In: *IEEE Transactions on Neural Net–works* 11.3, pp. 601–614.

Aly, Mohamed, Welinder, Peter, Munich, Mario, and Perona, Pietro (2009). “Automatic Discovery of Image Families: Global vs. Local Features”. In: *IEEE International Conference on Image Processing*, pp. 777–780.

Angeli, Adrien, Filliat, David, Doncieux, Stéphane, and Meyer, Jean-Arcady (2008). “Fast and Incremental Method for Loop-closure Detection Using Bags of Visual Words”. In: *IEEE Transactions on Robotics* 24.5, pp. 1027–1037.

Arandjelović, Relja, Gronat, Petr, Torii, Akihiko, Pajdla, Tomas, and Sivic, Josef (2018). “NetVLAD: CNN Architecture for Weakly Supervised Place Recognition”. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 40.6, pp. 1437–1451.

Arleo, Angelo and Gerstner, Wulfram (2000). “Spatial Cognition and Neuro-mimetic Nav–igation: A Model of Hippocampal Place Cell Activity”. In: *Biological Cybernetics* 83.3, pp. 287–299.

Arroyo, Roberto, Alcantarilla, Pablo F, Bergasa, Luis M, Yebes, J Javier, and Bronte, Sebastián (2014). “Fast and Effective Visual Place Recognition Using Binary Codes and Disparity Information”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3089–3094.

Artac, Matej, Jogan, Matjaz, and Leonardis, Ales (2002). “Incremental PCA for On-line Visual Learning and Recognition”. In: *Object Recognition Supported by User Interaction for Service Robots*, pp. 781–784.

Mur-Artal, Raúl and Tardós, Juan D. (2014). “Fast Relocalisation and Loop Closing in Keyframe-based SLAM”. In: *IEEE International Conference on Robotics and Automation*, pp. 846–853.

Baeza-Yates, Ricardo A. and Ribeiro-Neto, Berthier (1999). *Modern information retrieval*. Addison-Wesley Longman Publishing Co., Inc.

Bailey, Tim (2003). “Constrained Initialisation for Bearing-only SLAM”. In: *IEEE International Conference on Robotics and Automation*, pp. 1966–1971.

Bailey, Tim and Durrant-Whyte, Hugh (2006). “Simultaneous Localization and Mapping (SLAM): Part II”. In: *IEEE Robotics & Automation Magazine* 13.3, pp. 108–117.

Ball, David, Heath, Scott, Wiles, Janet, et al. (2013). “OpenRatSLAM: An Open Source Brain-based SLAM System”. In: *Autonomous Robots* 34.3, pp. 149–176.

Barrera, Alejandra and Weitzenfeld, Alfredo (2008). “Biologically-inspired Robot Spatial Cognition based on Rat Neurophysiological Studies”. In: *Autonomous Robots* 25.1-2, pp. 147–169.

Bay, Herbert, Ess, Andreas, Tuytelaars, Tinne, and Van Gool, Luc (2008). “Speeded-up Robust Features (SURF)”. In: *Computer Vision and Image Understanding* 110.3, pp. 346–359.

Bekris, Kostas E, Glick, Max, and Kavraki, Lydia E (2006). “Evaluation of Algorithms for Bearing-only SLAM”. In: *IEEE International Conference on Robotics and Automation*, pp. 1937–1943.

Best, Phillip J., White, Aaron M., and Minai, Ali (2001). “Spatial Processing in the Brain: The Activity of Hippocampal Place Cells”. In: *Annual Review of Neuroscience* 24.1, pp. 459–486.

Carlevaris-Bianco, Nicholas and Eustice, Ryan M. (2014). “Learning Visual Feature Descriptors for Dynamic Lighting Conditions”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems IROS*, pp. 2769–2776.

Biederman, Irving (1995). *Visual object recognition*. Vol. 2. MIT press Cambridge, MA.

Biswas, Hridkamol, Kazmi, S. M. Ali Musa, and Mertsching, Bärbel (2019). “Learning Condition-invariant Scene Representations for Across the Seasons Place Recognition”.

Blanco, Jose-Luis, Moreno, Francisco-Angel, and Gonzalez, Javier (2009). “A Collection of Outdoor Robotic Datasets with Centimeter-accuracy Ground Truth”. In: *Autonomous Robots* 27.4, p. 327.

Boehm, Richard G. and Petersen, James F. (1994). “An Elaboration of the Fundamental Themes in Geography”. In: *Social Education* 58.4, pp. 211–18.

University of Bristol (2011). *Memory Systems*. https://www.bristol.ac.uk/synaptic/basics/basics-7.html. Accessed: 2024-07-10.

Buschman, Timothy J and Miller, Earl K (2007). “Top-down versus bottom-up control of attention in the prefrontal and posterior parietal cortices”. In: *Science* 315.5820, pp. 1860–1862.

Calonder, Michael, Lepetit, Vincent, Strecha, Christoph, and Fua, Pascal (2010). “BRIEF: Binary Robust Independent Elementary Features”. In: *Computer Vision – ECCV 2010*, pp. 778–792.

Canziani, Alfredo, Paszke, Adam, and Culurciello, Eugenio (2016). “An Analysis of Deep Neural Network Models for Practical Applications”. In: *arXiv:1605.07678*.

Cattaneo, Daniele, Vaghi, Matteo, and Valada, Abhinav (2021). “LCDNet: Deep Loop Closure Detection for LiDAR SLAM based on Unbalanced Optimal Transport”. In: *arXiv preprint arXiv:2103.05056*.

Chen, Zetao, Lam, Obadiah, Jacobson, Adam, and Milford, Michael (2014). “Convolutional Neural Network-based Place Recognition”. In: *Australasian Conference on Robotics and Automation*.

Chen, Zetao, Lowry, Stephanie, Jacobson, Adam, Ge, Zongyuan, and Milford, Michael (2015). “Distance Metric Learning for Feature-agnostic Place Recognition”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2556–2563.

Chen, Liang-Chieh, Papandreou, George, Schroff, Florian, and Adam, Hartwig (2017). "Rethinking Atrous Convolution for Semantic Image Segmentation". In: *arXiv:1706.05587*.

Chinchor, Nancy (1992). "MUC-4 evaluation metrics". In: *Proceedings of the 4th Conference on Message Understanding*, pp. 22–29.

Choset, Howie and Nagatani, Keiji (2001). "Topological Simultaneous Localization and Mapping (SLAM): Toward Exact Localization without Explicit Localization". In: *IEEE Transactions on Robotics and Automation* 17.2, pp. 125–137.

Civera, Javier, Davison, Andrew J., and Montiel, J. M. Martinez (2008). "Inverse Depth Parametrization for Monocular SLAM". In: *IEEE Transactions on Robotics* 24.5, pp. 932–945.

Clemente, Laura A., Davison, Andrew J., Reid, Ian D., Neira, José, and Tardós, Juan D. (2007). "Mapping Large Loops with a Single Hand-held Camera". In: *Robotics: Science and Systems*. Vol. 2. 2.

Connor, Charles E, Egeth, Howard E, and Yantis, Steven (2004). "Visual attention: bottom-up versus top-down". In: *Current biology* 14.19, R850–R852.

Cummins, Mark and Newman, Paul (2008). "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance". In: *The International Journal of Robotics Research* 27.6, pp. 647–665.

Cummins, Mark and Newman, Paul (2011). "Appearance-only SLAM at Large Scale with FAB-MAP 2.0". In: *The International Journal of Robotics Research* 30.9, pp. 1100–1123.

Dalal, Navneet and Triggs, Bill (2005). "Histograms of Oriented Gradients for Human Detection". In: *IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 886–893.

Davison, Andrew J., Reid, Ian D., Molton, Nicholas D., and Stasse, Olivier (2007). "MonoSLAM: Real-time Single Camera SLAM". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.6, pp. 1052–1067.

Dellaert, Frank and Kaess, Michael (2006). "Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing". In: *The International Journal of Robotics Research* 25.12, pp. 1181–1203.

Dissanayake, MWM Gamini, Newman, Paul, Clark, Steve, Durrant-Whyte, Hugh F., and Csorba, Michael (2001). "A Solution to the Simultaneous Localization and Map Building (SLAM) Problem". In: *IEEE Transactions on Robotics and Automation* 17.3, pp. 229–241.

Doucet, Arnaud, Freitas, Nando de, Murphy, Kevin, and Russell, Stuart (2000). "Rao-black–wellised Particle Filtering for Dynamic Bayesian Networks". In: *International Conference on Uncertainty in Artificial Intelligence*, pp. 176–183.

Einstein, Albert and Shaw, George Bernard (2012). *Einstein on cosmic religion and other opinions and aphorisms*. Courier Corporation.

Fei-Fei, Li, Iyer, Asha, Koch, Christof, and Perona, Pietro (2007). "What Do We Perceive in a Glance of a Real-world Scene?" In: *Journal of vision* 7.1, pp. 10–10.

Ferrarini, Bruno, Milford, Michael, McDonald-Maier, Klaus D, and Ehsan, Shoaib (2022). "Highly-efficient binary neural networks for visual place recognition". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5493–5500.

Garcia-Fidalgo, Emilio and Ortiz, Alberto (2014). “On the Use of Binary Feature Descriptors for Loop Closure Detection”. In: *IEEE Emerging Technology and Factory Automation*, pp. 1–8.

Garcia-Fidalgo, Emilio and Ortiz, Alberto (2015). “Vision-based Topological Mapping and Localization Methods: A Survey”. In: *Robotics and Autonomous Systems* 64, pp. 1–20.

Garcia-Fidalgo, Emilio and Ortiz, Alberto (2017). “Hierarchical Place Recognition for Topological Mapping”. In: *IEEE Transactions on Robotics* 33.5, pp. 1061–1074.

Garcia-Fidalgo, Emilio and Ortiz, Alberto (2018). “iBoW-LCD: An Appearance-based Loop Closure Detection Approach Using Incremental Bags of Binary Words”. In: *IEEE Robotics and Automation Letters* 3.4, pp. 3051–3057.

Filliat, David (2007). “A Visual Bag of Words Method for Interactive Qualitative Localization and Mapping”. In: *IEEE International Conference on Robotics and Automation*, pp. 3921–3926.

Fischler, Martin A. and Bolles, Robert C. (1981). “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography”. In: *Communications of the ACM* 24.6, pp. 381–395.

Fox, Dieter, Burgard, Wolfram, Dellaert, Frank, and Thrun, Sebastian (1999). “Monte Carlo Localization: Efficient Position Estimation for Mobile Robots”. In: *AAAI*, pp. 343–349.

Fraundorfer, Friedrich and Scaramuzza, Davide (2012). “Visual Odometry: Part II: Matching, Robustness, Optimization, and Applications”. In: *IEEE Robotics Automation Magazine* 19.2, pp. 78–90.

Fritzke, Bernd (1997). “Some Competitive Learning Methods”. In: *Artificial Intelligence Institute, Dresden University of Technology*.

Garg, Sourav and Milford, Michael (2021). “SeqNet: Learning Descriptors for Sequence-based Hierarchical Place Recognition”. In: *arXiv preprint arXiv:2102.11603*.

Gehrig, Mathias, Stumm, Elena, Hinzmann, Timo, and Siegwart, Roland (2017). “Visual Place Recognition with Probabilistic Voting”. In: *IEEE International Conference on Robotics and Automation*, pp. 3192–3199.

Geiger, Andreas, Lenz, Philip, and Urtasun, Raquel (2012). “Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite”. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361.

GET Lab (2016). *Simultaneous Place Learning and Recognition*. https://www.youtube.com/watch?v=APRmam5XK24. Accessed: 2024-07-10.

Gibson, James Jerome (1966). “The Senses Considered as Perceptual Systems”. In: *Houghton Mifflin*.

Glover, Arren J., Maddern, William P., Milford, Michael J., and Wyeth, Gordon Fraser (2010). “FAB-MAP+RatSLAM: Appearance-based SLAM for Multiple Times of Day”. In: *IEEE International Conference on Robotics and Automation*, pp. 3507–3512.

Goldstein, E. Bruce (2011). *Cognitive Psychology: Connecting Mind, Research, and Everyday Experience*. Wadsworth Cengage Learning.

Goldstein, E. Bruce (2014). *Sensation and perception*. Wadsworth, Cengage Learning.

Golfarelli, M., Maio, D., and Rizzi, S. (1998). “Elastic Correction of Dead-reckoning Errors in Map Building”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 905–911.

Gregory, Richard L. (1997). “Knowledge in Perception and Illusion”. In: *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* 352.1358, pp. 1121–1127.

Grisetti, Giorgio, Stachniss, Cyrill, and Burgard, Wolfram (2009). “Nonlinear Constraint Network Optimization for Efficient Map Learning”. In: *IEEE Transactions on Intelligent Transportation Systems* 10.3, pp. 428–439.

Grisetti, Giorgio, Kümmerle, Rainer, Stachniss, Cyrill, and Burgard, Wolfram (2010a). “A Tutorial on Graph-based SLAM”. In: *IEEE Intelligent Transportation Systems Magazine* 2.4, pp. 31–43.

Grisetti, Giorgio, Kümmerle, Rainer, Stachniss, Cyrill, Frese, Udo, and Hertzberg, Christoph (2010b). “Hierarchical Optimization on Manifolds for Online 2D and 3D Mapping”. In: *IEEE International Conference on Robotics and Automation*, pp. 273–278.

Gutmann, J.-S. and Konolige, Kurt (1999). “Incremental Mapping of Large Cyclic Environments”. In: *International Symposium on Computational Intelligence in Robotics and Automation*. IEEE, pp. 318–325.

Hafner, Verena Vanessa (2000). “Learning Places in Newly Explored Environments”. In: *in Meyer, Berthoz, Floreano, Roitblat and Wilson (Eds.), SAB2000 Proceedings Supplement Book, Publication of the International Society for Adaptive Behavior*.

Hafting, Torkel, Fyhn, Marianne, Molden, Sturla, Moser, May-Britt, and Moser, Edvard I. (2005). “Microstructure of a Spatial Map in the Entorhinal Cortex”. In: *Nature* 436.7052, pp. 801–806.

Hamze, F. and Clark, J. J. (2001). “View-based Route-learning with Self-organizing Neural Networks”. In: *Image and Vision Computing* 19.11, pp. 753–761.

Han, Fei, Yang, Xue, Deng, Yiming, et al. (2017). “SRAL: Shared Representative Appearance Learning for Long-term Visual Place Recognition”. In: *IEEE Robotics and Automation Letters* 2.2, pp. 1172–1179.

Hartley, Tom, Lever, Colin, Burgess, Neil, and O’Keefe, John (2013). “Space in the Brain: How the Hippocampal Formation Supports Spatial Cognition”. In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 369.1635, pp. 20120510–20120510.

Henry, Peter, Krainin, Michael, Herbst, Evan, Ren, Xiaofeng, and Fox, Dieter (2012). “RGB-D Mapping: Using Kinect-Style Depth Cameras for Dense 3D Modeling of Indoor Environments”. In: *The International Journal of Robotics Research* 31.5, pp. 647–663.

Hou, Yi, Zhang, Hong, and Zhou, Shilin (2015). “Convolutional Neural Network-based Image Representation for Visual Loop Closure Detection”. In: *2015 IEEE International Conference on Information and Automation*, pp. 2238–2245.

Jaakkola, Tommi and Haussler, David (1999). “Exploiting Generative Models in Discriminative Classifiers”. In: *Adv. Neural Inf. Process. Syst.* Pp. 487–493.

Jacobson, Adam, Chen, Zetao, and Milford, Michael (2015). “Online Place Recognition Calibration for Out-of-the-box SLAM”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1357–1364.

Jégou, Hervé, Douze, Matthijs, Schmid, Cordelia, and Pérez, Patrick (2010). "Aggregating Local Descriptors into a Compact Image Representation". In: *IEEE Int. Conf. Comput. Vis. Pattern Recognit.* Pp. 3304–3311.

Jia, Yangqing, Shelhamer, Evan, Donahue, Jeff, et al. (2014). "Caffe: Convolutional Architecture for Fast Feature Embedding". In: *Proceedings of the 22nd ACM International Conference on Multimedia*. ACM, pp. 675–678.

Johns, Edward and Yang, Guang-Zhong (2014). "Generative Methods for Long-Term Place Recognition in Dynamic Scenes". In: *International Journal of Computer Vision* 106.3, pp. 297–314.

Kaess, M., Ranganathan, A., and Dellaert, F. (2008). "iSAM: Incremental Smoothing and Mapping". In: *IEEE Transactions on Robotics* 24.6, pp. 1365–1378.

Kawewong, Aram, Tongprasit, Noppharit, Tangruamsub, Sirinart, and Hasegawa, Osamu (2011). "Online and Incremental Appearance-based SLAM in Highly Dynamic Environments". In: *The International Journal of Robotics Research* 30.1, pp. 33–55.

Kazmi, S. M. Ali Musa, Rao, Naveed Iqbal, Fazal, Fawad, and Khan, Muhammad Faisal (2012). "Exploiting a Scene Calibration Mechanism for Depth Estimation". In: *International Conference on Image Processing Theory, Tools and Applications*, pp. 421–425.

Kazmi, S. M. Ali Musa and Mertsching, Bärbel (2016a). "Gist+RatSLAM: An Incremental Bio-inspired Place Recognition Front-end for RatSLAM". In: *9th International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, pp. 27–34.

Kazmi, S. M. Ali Musa and Mertsching, Bärbel (2016b). "Simultaneous Place Learning and Recognition for Real-time Appearance-based Mapping". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4898–4903.

Kazmi, S. M. Ali Musa and Mertsching, Bärbel (2019a). "Detecting the Expectancy of a Place Using Nearby Context for Appearance-based Mapping". In: *IEEE Transactions on Robotics*.

Kazmi, S. M. Ali Musa, Mohamed, Mahmoud A., and Mertsching, Bärbel (2019b). "Feature-agnostic Low-cost Place Recognition for Appearance-based Mapping". In: *12th International Conference on Computer Vision*.

Khan, Sheraz and Wollherr, Dirk (2015). "IBuILD: Incremental Bag of Binary Words for Appearance based Loop Closure Detection". In: *IEEE International Conference on Robotics and Automation*, pp. 5441–5447.

Knopp, Jan, Sivic, Josef, and Pajdla, Tomas (2010). "Avoiding Confusing Features in Place Recognition". In: *European Conference on Computer Vision*, pp. 748–761.

Kohonen, Teuvo (1990). "The Self-organizing Map". In: *Proceedings of the IEEE* 78.9, pp. 1464–1480.

Kohonen, Teuvo (1997). *Self-Organizing Maps*. Springer series in information sciences 30. Springer.

Konolige, Kurt (2004). "Large-scale Map-making". In: *AAAI*, pp. 457–463.

Konolige, Kurt, Grisetti, Giorgio, Kümmerle, Rainer, et al. (2010). "Efficient Sparse Pose Adjustment for 2D Mapping". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 22–29.

Kosecka, Jana, Zhou, Liang, Barber, Philip, and Duric, Zoran (2003). "Qualitative Image based Localization in Indoors Environments". In: *IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 3–8.

Krajník, Tomáš, Cristóforis, Pablo, Kusumam, Keerthy, Neubert, Peer, and Duckett, Tom (2017). "Image Features for Visual Teach-and-repeat Navigation in Changing Environments". In: *Robotics and Autonomous Systems* 88, pp. 127–141.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*, pp. 1097–1105.

Krose, B.J.A. and Bunschoten, Roland (1999). "Probabilistic Localization by Appearance Models and Active Vision". In: *IEEE International Conference on Robotics and Automation*, pp. 2255–2260.

Kuipers, Benjamin and Byun, Yung-Tai (1988). "A Robust, Qualitative Method for Robot Spatial Learning". In: *AAAI*, pp. 774–779.

Kümmerle, Rainer, Grisetti, Giorgio, Strasdat, Hauke, Konolige, Kurt, and Burgard, Wolfram (2011). "G2o: A General Framework for Graph Optimization". In: *IEEE International Conference on Robotics and Automation*, pp. 3607–3613.

Lategahn, H., Beck, J., Kitt, B., and Stiller, C. (2013). "How to Learn an Illumination Robust Image Feature for Place Recognition". In: *IEEE Intelligent Vehicles Symposium (IV)*, pp. 285–291.

Li, Xiaowei, Wu, Changchang, Zach, Christopher, Lazebnik, Svetlana, and Frahm, Jan-Michael (2008). "Modeling and Recognition of Landmark Image Collections Using Iconic Scene Graphs". In: *European Conference on Computer Vision*, pp. 427–440.

Gálvez-López, Dorian and Tardós, Juan D. (2012). "Bags of Binary Words for Fast Place Recognition in Image Sequences". In: *IEEE Transactions on Robotics* 28.5, pp. 1188–1197.

Lowe, David G. (2004). "Distinctive Image Features from Scale-invariant Keypoints". In: *International Journal of Computer Vision* 60.2, pp. 91–110.

Lowry, Stephanie and Milford, Michael J. (2016a). "Supervised and Unsupervised Linear Learning Techniques for Visual Place Recognition in Changing Environments". In: *IEEE Transactions on Robotics* 32.3, pp. 600–613.

Lowry, Stephanie, Sünderhauf, Niko, Newman, Paul, et al. (2016b). "Visual Place Recognition: A Survey". In: *IEEE Transactions on Robotics* 32.1, pp. 1–19.

Lu, Feng and Milios, Evangelos (1997). "Globally Consistent Range Scan Alignment for Environment Mapping". In: *Autonomous Robots* 4.4, pp. 333–349.

Luo, Jie, Pronobis, Andrzej, Caputo, Barbara, and Jensfelt, Patric (2007). "Incremental Learning for Place Recognition in Dynamic Environments". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 721–728.

Marr, David and Hildreth, Ellen (1980). "Theory of Edge Detection". In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 207.1167, pp. 187–217.

McManus, Colin, Upcroft, Ben, and Newman, Paul (2015). "Learning Place-dependant Features for Long-term Vision-based Localisation". In: *Autonomous Robots* 39.3, pp. 363–387.

Mikkulainen, Risto (2005). *Computational Maps in the Visual Cortex*. Springer.

Milford, Michael, Prasser, David, and Wyeth, Gordon (2005). "Experience Mapping: Producing Spatially Continuous Environment Representations Using RatSLAM". In: *Proceedings of Australasian Conference on Robotics and Automation*.

Milford, Michael and Wyeth, Gordon (2008). "Mapping a Suburb with a Single Camera Using a Biologically Inspired SLAM System". In: *IEEE Transactions on Robotics* 24.5, pp. 1038–1053.

Milford, Michael J. and Wyeth, Gordon Fraser (2012). "SeqSLAM: Visual Route-based Navigation for Sunny Summer Days and Stormy Winter Nights". In: *IEEE International Conference on Robotics and Automation*, pp. 1643–1649.

Mirabdollah, Mohammad Hossein, Mertsching, Bärbel, and Häb-Umbach, Reinhold (2016). "Robust Techniques for Monocular Simultaneous Localization and Mapping". PhD thesis.

Mokssit, Saad, Licea, Daniel Bonilla, Guermah, Bassma, and Ghogho, Mounir (2023). "Deep Learning Techniques for Visual SLAM: A Survey". In: *IEEE Access* 11, pp. 20026–20050.

Montemerlo, Michael, Thrun, Sebastian, Koller, Daphne, and Wegbreit, Ben (2002). "FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem". In: *AAAI*, pp. 593–598.

Montemerlo, Michael (2007). *FastSLAM: A Scalable Method for the Simultaneous Localization and Mapping Problem in Robotics*. Springer.

Murillo, Ana C, Singh, Gautam, Kosecká, Jana, and Guerrero, José Jesús (2013). "Localization in Urban Environments Using a Panoramic Gist Descriptor". In: *IEEE Transactions on Robotics* 29.1, pp. 146–160.

Nair, Vinod and Hinton, Geoffrey E. (2010). "Rectified Linear Units Improve Restricted Boltzmann Machines". In: *27th International Conference on Machine Learning*, pp. 807–814.

Naseer, Tayyab, Ruhnke, Michael, Stachniss, Cyrill, Spinello, Luciano, and Burgard, Wolfram (2015). "Robust Visual SLAM Across Seasons". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2529–2535.

Naseer, Tayyab, Suger, Benjamin, Ruhnke, Michael, and Burgard, Wolfram (2017b). "Vision-based Markov Localization for Long-term Autonomy". In: *Robotics and Autonomous Systems* 89, pp. 147–157.

Naseer, T., Oliveira, G. L., Brox, T., and Burgard, W. (2017a). "Semantics-Aware Visual Localization under Challenging Perceptual Conditions". In: *IEEE International Conference on Robotics and Automation*, pp. 2614–2620.

Neubert, Peer, Sünderhauf, Niko, and Protzel, Peter (2015). "Superpixel-based Appearance Change Prediction for Long-term Navigation Across Seasons". In: *Robotics and Autonomous Systems* 69, pp. 15–27.

Neubert, Peer and Protzel, Peter (2015). "Local Region Detector+CNN based Landmarks for Practical Place Recognition in Changing Environments". In: *IEEE European Conference on Mobile Robots*, pp. 1–6.

Nicosevici, Tudor and Garcia, Rafael (2012). "Automatic Visual Bag-of-Words for Online Robot Navigation and Mapping". In: *IEEE Transactions on Robotics* 28.4, pp. 886–898.

Nistér, David (2004). “An Efficient Solution to the Five-point Relative Pose Problem”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.6, pp. 0756–777.

Nistér, David, Naroditsky, Oleg, and Bergen, James (2006). “Visual Odometry for Ground Vehicle Applications”. In: *Journal of Field Robotics* 23.1, pp. 3–20.

O’Keefe, John and Nadel, Lynn (1978). *The Hippocampus as a Cognitive map*. Oxford University Press.

O’Keefe, John (1979). “A Review of the Hippocampal Place Cells”. In: *Progress in Neurobiology* 13.4, pp. 419–439.

Oliva, Aude and Torralba, Antonio (2001). “Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope”. In: *International Journal of Computer Vision* 42.3, pp. 145–175.

Oliva, Aude and Torralba, Antonio (2006). “Building the Gist of a Scene: The Role of Global Image Features in Recognition”. In: *Progress in Brain Research* 155, pp. 23–36.

Oliva, Aude, Torralba, Antonio, Guérin-Dugué, Anne, and Hérault, Jeanny (1999). “Global Semantic Classification of Scenes Using Power Spectrum Templates”. In: *International Conference on Challenge of Image Retrieval*. British Computer Society, pp. 9–9.

Oliveira, Gabriel L., Burgard, Wolfram, and Brox, Thomas (2016). “Efficient Deep Models for Monocular Road Segmentation”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4885–4891.

Olson, Edwin, Leonard, John, and Teller, Seth (2006). “Fast Iterative Alignment of Pose Graphs with Poor Initial Estimates”. In: *IEEE International Conference on Robotics and Automation*, pp. 2262–2269.

Pepperell, Edward, Corke, Peter I., and Milford, Michael J. (2014). “All-environment Visual Place Recognition with SMART”. In: *IEEE International Conference on Robotics and Automation*, pp. 1612–1618.

Pepperell, Edward, Corke, Peter, and Milford, Michael (2016). “Routed Roads: Probabilistic Vision-based Place Recognition for Changing Conditions, Split Streets and Varied Viewpoints”. In: *The International Journal of Robotics Research* 35.9, pp. 1057–1179.

Potter, Mary C. (1976). “Short-term Conceptual Memory for Pictures”. In: *Journal of Experimental Psychology: Human Learning and Memory* 2.5, pp. 509–522.

Ranganathan, Ananth, Menegatti, Emanuele, and Dellaert, Frank (2006). “Bayesian Inference in the Space of Topological Maps”. In: *IEEE Transactions on Robotics* 22.1, pp. 92–107.

Ranganathan, Ananth, Matsumoto, Shinichi, and Ilstrup, David (2013). “Towards Illumination Invariance for Visual Localization”. In: *IEEE International Conference on Robotics and Automation*, pp. 3791–3798.

RatSLAM (2009). *Robotics@QUT-St Lucia Loop-test Dataset*. https://wiki.qut.edu.au/display/cyphy/RatSLAM. Accessed: 2024-07-10.

Rebai, Karima, Azouaoui, Ouahiba, and Achour, Nouara (2013). “Fuzzy ART-based Place Recognition for Visual Loop Closure Detection”. In: *Biological Cybernetics* 107.2, pp. 247–259.

Rijsbergen, C. J. Van (1979). *Information Retrieval*. Butterworth-Heinemann.

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). “ORB: An Efficient Alternative to SIFT or SURF”. In: *IEEE International Conference on Computer Vision*, pp. 2564–2571.

Scaramuzza, Davide and Fraundorfer, Friedrich (2011). “Visual Odometry: Part I: The First 30 Years and Fundamentals”. In: *IEEE Robotics Automation Magazine* 18.4, pp. 80–92.

Scherer, Dominik, Müller, Andreas, and Behnke, Sven (2010). “Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition”. In: *International Conference on Artificial Neural Networks*, pp. 92–101.

Schönberger, Johannes L., Pollefeys, Marc, Geiger, Andreas, and Sattler, Torsten (2017). “Semantic Visual Localization”. In: *arXiv:1712.05773*.

Schubert, Stefan, Neubert, Peer, Garg, Sourav, Milford, Michael, and Fischer, Tobias (2023). “Visual Place Recognition: A Tutorial”. In: *arXiv preprint arXiv:2303.03281*.

Schyns, Philippe G. and Oliva, Aude (1994). “From Blobs to Boundary Edges: Evidence for Time- and Spatial-scale-dependent Scene Recognition”. In: *Psychological Science* 5.4, pp. 195–200.

Sermanet, Pierre, Eigen, David, Zhang, Xiang, et al. (2014). “Overfeat: Integrated Recognition, Localization and Detection Using Convolutional Networks”. In: *International Conference on Learning Representations*.

Shi, Jianbo and Tomasi, Carlo (1994). “Good Features to Track”. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 593–600.

Siagian, Christian and Itti, Laurent (2009). “Biologically Inspired Mobile Robot Vision Localization”. In: *IEEE Transactions on Robotics* 25.4, pp. 861–873.

Siam, S. M. and Zhang, H. (2017). “Fast-SeqSLAM: A Fast Appearance based Place Recognition Algorithm”. In: *IEEE International Conference on Robotics and Automation*, pp. 5702–5708.

Siciliano, Bruno and Khatib, Oussama, eds. (2008). *Springer Handbook of Robotics*. Springer-Verlag.

Sivic, Josef and Zisserman, Andrew (2003). “Video Google: A Text Retrieval Approach to Object Matching in Videos”. In: *IEEE International Conference on Computer Vision*, pp. 1470–1477.

Smith, Randall C. and Cheeseman, Peter (1986). “On the Representation and Estimation of Spatial Uncertainty”. In: *The International Journal of Robotics Research* 5.4, pp. 56–68.

Smith, Randall, Self, Matthew, and Cheeseman, Peter (1990). “Estimating Uncertain Spatial Relationships in Robotics”. In: *Autonomous Robot Vehicles*. Springer, pp. 167–193.

Sünderhauf, Niko and Protzel, Peter (2011). “BRIEF-Gist-Closing the Loop by Simple Means”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1234–1241.

Sünderhauf, Niko, Neubert, Peer, and Protzel, Peter (2013). “Are We There Yet? Challenging SeqSLAM on a 3000 km Journey Across All Four Seasons”. In: *Workshop on Long-term Autonomy, IEEE Int. Conf. Robot. Autom.*

Sünderhauf, Niko, Shirazi, Sareh, Dayoub, Feras, Upcroft, Ben, and Milford, Michael (2015a). “On the Performance of ConvNet Features for Place Recognition”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4297–4304.

Sünderhauf, Niko, Shirazi, Sareh, Jacobson, Adam, et al. (2015b). “Place Recognition with ConvNet Landmarks: Viewpoint-robust, Condition-robust, Training-free”. In: *Proceedings of Robotics: Science and Systems XII*.

Taube, Jeffrey S., Muller, Robert U., and Ranck, James B. (1990). “Head-direction Cells Recorded from the Postsubiculum in Freely Moving Rats. I. Description and Quantitative Analysis”. In: *The Journal of Neuroscience* 10.2, pp. 420–435.

Thrun, Sebastian, Burgard, Wolfram, and Fox, Dieter (2005). *Probabilistic Robotics*. MIT press.

Thrun, Sebastian and Montemerlo, Michael (2006). “The Graph SLAM Algorithm with Applications to Large-scale Mapping of Urban Structures”. In: *The International Journal of Robotics Research* 25.5-6, pp. 403–429.

Torii, Akihiko, Arandjelovic, Relja, Sivic, Josef, Okutomi, Masatoshi, and Pajdla, Tomas (2015). “24/7 Place Recognition by View Synthesis”. In: *IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 1808–1817.

Tsintotas, Konstantinos A., Bampis, Loukas, and Gasteratos, Antonios (2022). “The Revisiting Problem in Simultaneous Localization and Mapping: A Survey on Visual Loop Closure Detection”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.11, pp. 19929–19953.

Turcot, Panu and Lowe, David G. (2009). “Better Matching with Fewer Features: The Selection of Useful Features in Large Database Recognition Problems”. In: *IEEE International Conference on Computer Vision Workshops*, pp. 2109–2116.

Ulrich, Iwan and Nourbakhsh, Illah (2000). “Appearance-based Place Recognition for Topological Localization”. In: *IEEE International Conference on Robotics and Automation*, pp. 1023–1029.

Valgren, Christoffer and Lilienthal, Achim J. (2010). “SIFT, SURF & Seasons: Appearance-based Long-term Localization in Outdoor Environments”. In: *Robotics and Autonomous Systems* 58.2, pp. 149–156.

Van Der Merwe, Rudolph, Doucet, Arnaud, De Freitas, Nando, and Wan, Eric A (2001). “The Unscented Particle Filter”. In: *Advances in Neural Information Processing Systems*, pp. 584–590.

Vassallo, R.F., Santos-Victor, J., and Schneebeli, H.J. (2002). “Using Motor Representations for Topological Mapping and Navigation”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 478–483.

Wang, H., Wang, C., and Xie, L. (2021). “Intensity-SLAM: Intensity Assisted Localization and Mapping for Large Scale Environment”. In: *IEEE Robotics and Automation Letters*.

Williams, Brian, Cummins, Mark, Neira, José, et al. (2008). “An Image-to-map Loop Closing Method for Monocular SLAM”. In: pp. 2053–2059.

Williams, Brian, Cummins, Mark, Neira, José, et al. (2009). “A Comparison of Loop Closing Techniques in Monocular SLAM”. In: *Robotics and Autonomous Systems* 57.12, pp. 1188–1197.

Wu, Junjun, Zhang, Hong, and Guan, Yisheng (2014). “An Efficient Visual Loop Closure Detection Method in a Map of 20 Million Key Locations”. In: *IEEE International Conference on Robotics and Automation*, pp. 861–866.

Wyeth, Gordon and Milford, Michael (2009). “Spatial Cognition for Robots”. In: *IEEE Robotics & Automation Magazine* 16.3, pp. 24–32.

Xiang, J. Z. and Brown, M. W. (1998). “Differential Neuronal Encoding of Novelty, Familiarity and Recency in Regions of the Anterior Temporal Lobe”. In: *Neuropharmacology* 37.4–5, pp. 657–676.

Xu, X., Yin, H., Chen, Z., et al. (2021). “DiSCO: Differentiable Scan Context With Orientation”. In: *IEEE Robotics and Automation Letters* 6.2, pp. 2791–2798.

Yeh, Tom, Lee, John, and Darrell, Trevor (2007). “Adaptive Vocabulary Forests by Dynamic Indexing and Category Learning”. In: *IEEE International Conference on Computer Vision*, pp. 1–8.

Yousif, Khalid, Bab-Hadiashar, Alireza, and Hoseinnezhad, Reza (2015). “An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics”. In: *Intelligent Industrial Systems* 1.4, pp. 289–311.

Yu, X., Chaturvedi, S., Feng, C., et al. (2018). “VLASE: Vehicle Localization by Aggregating Semantic Edges”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3196–3203.

Zhang, Guangcong, Lilly, Mason J, and Vela, Patricio A (2016). “Learning Binary Features Online from Motion Dynamics for Incremental Loop-closure Detection and Place Recognition”. In: *IEEE International Conference on Robotics and Automation*, pp. 765–772.

Zhou, Bolei, Lapedriza, Agata, Xiao, Jianxiong, Torralba, Antonio, and Oliva, Aude (2014). “Learning Deep Features for Scene Recognition Using Places Database”. In: *Advances in Neural Information Processing Systems*, pp. 487–495.

List of Publications

[1] Kazmi, S. M. Ali Musa and Mertsching, Bärbel (2019a). "*Detecting the Expectancy of a Place Using Nearby Context for Appearance-based Mapping*". In: IEEE Transactions on Robotics (T-RO).

[2] Kazmi, S. M. Ali Musa, Mohamed, Mahmoud A. and Mertsching, Bärbel (2019b). "*Feature-agnostic Low-cost Place Recognition for Appearance-based Mapping*". In: 12th International Conference on Computer Vision (ICVS), Thessaloniki, Greece.

[3] Kazmi, S. M. Ali Musa and Mertsching, Bärbel (2016b). "*Simultaneous Place Learning and Recognition for Real-time Appearance-based Mapping*". In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea.

[4] Kazmi, S. M. Ali Musa and Mertsching, Bärbel (2016a). "*Gist+RatSLAM: An Incremental Bio-inspired Place Recognition Front-end for RatSLAM*". In: 9th International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), New York, USA.

[5] Kazmi, S. M. Ali Musa and Rao, Naveed Iqbal and Fazal, Fawad and Khan, Muhammad Faisal (2012). "*Exploiting a Scene Calibration Mechanism for Depth Estimation*". In: International Conference on International Conference on Image Processing Theory, Tools and Applications (IEEE), Istanbul, Turkey.

List of Abbreviations

$\sqrt{\text{SAM}}$	Square root simultaneous smoothing and mapping
3D	Three dimensions
AlexNet	ConvNet named after the author's name Alex
ANNS	Approximate nearest neighbor search
BoVW	Bag-of-visual-words
BoW	Bag-of-words
BRIEF	Binary robust independent elementary features
BRISK	Binary robust invariant scalable keypoints
Caffe	Convolutional architecture for fast feature embedding
CANN	Continuous attractor neural network
CC	CityCenter test sequence
CNN	Convolutional neural network
Conv1	First convolutional layer in ConvNet
Conv3	Third convolutional layer
ConvNet	Convolutional neural network

CPU	Central processing unit
CRFH	Composed receptive field histograms
DBoW	Dynamic bag of words
EEG	Electroencephalogram
EKF	Extended Kalman filter
FAB-MAP	Fast appearance-based mapping
FAST	Features from accelerated segment test
FC6	Sixth fully connected layer in ConvNet
FN	False negative
FOV	Field of view
FP	False positive
FRMI	Functional magnetic resonance imaging
Fuzzy ART	Fuzzy adaptive resonance theory
g2o	A general framework for graph optimization
GB	Gigabytes
GBN	Growing belief network
GHz	Gigahertz
GPS	Global position system
GPU	Graphics processing unit
GSF	GSOM-based front-end for place recognition
GSOM	Growing self-organizing maps

HMM	Hidden markov model
HOG	Histogram of oriented gradients
HOG-Man	Hierarchical optimization for pose-graphs on manifolds
iBoW-LCD	Incremental bag-of-words loop closure detection
IGBN	Improved growing belief network
iSAM	Incremental smoothing and mapping
kNN	k-nearest neighbors
KT##	KITTI test sequence number
L1-norm	First order vector normalization
L2-norm	Second order vector normalization
LBP	Local binary patterns
LDA	Linear discriminant analysis
LDB	Local difference binary features
LMNN	Large margin nearest neighbor
LSH	Locality-sensitive hashing
LVQ	Learning vector quantization
MAP	Maximum a posteriori
MLE	Maximum likelihood estimate
MP6L	Malaga parking lot with 6 lanes test sequence
NC	NewCollege test sequence
NCC	Normalized cross-correlation

NNS	Nearest neighbor search
ORB	Oriented FAST and rotated BRIEF
PCA	Principal component analysis
PHOG	Pyramid histogram of oriented gradients
PIRF	Position invariant robust features
Pool1	First pooling layer in ConvNet
PR	Precision–recall
QR	Matrix decomposition method
RANSAC	Random sample consensus
RatSLAM	A SLAM system inspired by navigation process in rats'
RBF	Radial basis function
ReLU	Rectified linear unit
RGB-D	Red, green, blue and depth channels
RMSE	Root mean square error
RSF	RatSLAM's front-end for place recognition
SeqSLAM	Sequence SLAM
SIFT	Scale invariant feature transform
SL08	St. Lucia test sequence 190809 (08:45)
SL12	St. Lucia test sequence 100909 (12:10)
SL14	St. Lucia test sequence 100909 (14:10)
SL15	St. Lucia test sequence 180809 (15:45)

SLAM	Simultaneous localization and mapping
SOM	Self-organizing maps
SRAL	Shared representative appearance learning
SSD	Sum of squared differences
SSE	Sum of squared error
SURF	Speeded up robust features
SVD	Singular value decomposition
SVM	Support vector machine
TN	True negative
TP	True positive
UKF	Unscented Kalman filter
V1	First area in the visual cortex where the signal arrives from eyes
V4	Fourth area in the hierarchy in a visual cortex
VO	Visual odometry

List of Notations

Alphabetical Symbols

a	Activation strength (i.e. output) of a neuron; or index of the 3D Gaussian mask in the RatSLAM
a_i	Binary activation of neuron i in GSF
$a(i^*)$	Activation strength of the maximizer neuron in GBN
b	Bias for adjusting the activation threshold of a neuron; or index of the 3D Gaussian mask in the RatSLAM
c	Index of the winner neuron; or index of the 3D Gaussian mask in the RatSLAM
D	Number of filters in a convolutional layer; or length of a feature descrip–tor
$d_e(p, q)$	Euclidean distance between vectors whose indexes are p and q
$d_s(p, q)$	Sum of squared differences between vectors whose indexes are p and q
E	Experience map in the RatSLAM system
e_i	i^{th} experience in the experience map
$f(.)$	Neuronal activation function; or a feature extractor function that computes the global description of the input image

F_β	Weighted average of the precision and recall coordinate pair
f_c	Spatial frequency, i.e., the number of cycles per image
$f_i^{(k)}$	Learning frequency of a neuron i at a time instant k
$\bar{\mathbf{g}}^{(t)}$	Visual description of the input image incorporating the nearby context
R_{fps}	Frame rate of a test sequence
$\mathbf{g}^{(k)}$	Gist feature vector of an input image $I^{(k)}$
$g_j^{(t)}$	j^{th} component of the gist features for image $I^{(t)}$
$G_i(u, v)$	Frequency response of the i^{th} transfer function
$g_i(x, y)$	Response of the i^{th} transfer function in spatial domain
$h(c, i)$	Scalar kernel that determines the connection strength between neurons
$H_i(u, v)$	An i^{th} transfer function (i.e. convolution mask) in frequency domain
$I^{(q)}$	Query image
i^*	Maximizer (i.e. highly active) neuron
$I^{(t)}$	An input image at a time instant t
$I(u, v)$	Image representation in frequency domain
$I(x, y)$	Image representation in spatial domain
K	Spatial size (width and height) of a kernel (aka filter)
k	A particular instance of a time such that $0 \leq k \leq t$
L	A random variable that instantiates to the maximizer neurons
l_k	A particular realization of the random variable L at time instant k
M	Number of landmarks in the map; or size of output matrix after convolution or pooling operation

$\mathbf{m}_j$	A vector denoting the position of the j^{th} landmark in the map
N_b	Number of blocks in an image
$N_c^{(t)}$	Neighborhood size around the winner at present time
N_h	Number of bins in a histogram
N	Number of rows (or columns) in an image; or number of neurons in a neural network
N_l	Number of sensed landmarks
N_p	Number of particles
$N^{(t)}$	Number of neurons in a GSOM network at present time
$n_{\theta'}$	Size of the pose cell network along θ' dimension
$n_{x'}$	Size of the pose cell network along x' dimension
$n_{y'}$	Size of the pose cell network along y' dimension
P	Probability mass function; or padding constant that adds zero rows and columns on image boundaries; or a pose network in the RatSLAM system
$\mathbf{p}^i$	Position of the i^{th} experience
P^i	Pose cells activity pattern stored in the i^{th} experience
$P_{x',y',\theta'}$	An element (x', y', θ') in the pose cell matrix
$\mathbf{S}$	A scatter matrix describing the covariance in predicated and observed landmark's positions
S	Length by which a filter is slid on an image
T	Total number of training iterations
t	Current time instant

t_0	Parameter that determines how far in time the maximizer neurons are observed to compute the expected network activity
u	Horizontal component of the spatial frequencies in Fourier domain
u_0	Filter center in frequency domain
$\mathbf{u}_k$	A column vector of the motion command (i.e. linear and angular velocities) at a time instant k
v	Linear velocity of a mobile robot; or vertical component of the spatial frequencies in Fourier domain
V	An array of view cells in the RatSLAM system
V^i	View cell stored in the i^{th} experience
V_i	i^{th} view cell in an array of view cells
w_i	i^{th} component of the neuron's weight vector
$\mathbf{w}_i^{(k)}$	Weight vector of the i^{th} neuron at an arbitrary time instant k
$w_{ji}^{(t)}$	j^{th} component of the i^{th} neuron's weight vector at a present time
$w_k^{(n)}$	Importance weight of the n^{th} particle at a time instant k
$\mathbf{w}_r^{(t)}$	Weight vector of the newly instantiated neuron
$\mathbf{W}^{(t)}$	Weight matrix at a time instant t
x	Horizontal component of the robot's position coordinate pair; or index of a horizontal coordinate in an image or a spatial filter
x_i	i^{th} component of the training example
$\mathbf{x}_j^{(t)}$	j^{th} input vector retrieved at a time instant t
$\mathbf{x}_k$	Robot's pose configuration $(x, y, \theta)^\top$ at a time instant k
$\mathbf{x}$	A training example

y	Vertical component of the robot's position coordinate pair; or index of a vertical coordinate in an image or a spatial filter
z	Weighted sum of inputs to a neuron
$\mathbf{z}_k^{(i)}$	Observed location of the i^{th} landmark at a time instant k in robot's frame
$\widetilde{\mathbf{z}}_t^{(i)}$	Predicted location of the i^{th} landmark at a time instant t in robot's frame

Greek Symbols

α	Current learning rate
α_0	Initial learning rate
α_f	Final learning rate
$\alpha_i^{(k)}$	Learning rate of neuron i at a time instant k
β	Bandwidth of the Gaussian RBF
$\beta_{i,x',y',\theta'}^{t+1}$	Connection strength between the view cell V_i and the pose cell $P_{x',y',\theta'}$
δ	Confidence threshold for decision of loop-closure detection
η	A normalization constant in Bayes rule; or error smoothness factor in GSOM
$\mathbb{E}$	Expectation operator
γ	Neuron creation weight; or smoothness level in exponential smoothing
λ	Number of Gabor filters in a bank; or learning constant in RatSLAM
$\mathcal{M}$	A set of point landmarks $\mathbf{m}_j : \forall j \in \{1, \ldots, M\}$ in the map
$\mathcal{M}^*$	A set of point landmarks representing optimal map of the environment
μ	Number of recently created neurons

$\boldsymbol{\mu}_t$	A mean vector that contains the robot pose and coordinate pairs of the landmarks' positions at time t
$\boldsymbol{\mu}_{k,m}^{(n)}$	A mean vector containing the position of the m^{th} landmark in the map maintained by n^{th} particle at a time instant k
$\mathcal{N}$	Multivariate normal distribution
$\Omega^{(t)}$	All place-to-neuron mappings up to present time
ω	A scalar quantity representing the rotational motion (angular velocity)
$\omega_i^{(k)}$	A set of places mapped to neuron i up to time instant k
$\Phi^{(t)}$	A set of network parameters at a time instant t; or no time superscript implies that the parameters do not evolve
ϕ	Scoring function that computes the similarity of two feature descriptors
π_m	Map (fixed) coordinate frame
π_r	Robot-centric coordinate frame
$\mathcal{S}^{(t)}$	State of the network at a time instant t
$\psi_{a,b,c}$	Inhibitory mask for a 3D Gaussian with negative weights
$p_{\mathcal{X}}$	A probability distribution defined over robot's pose configuration
$p_{\mathcal{X}\|\mathcal{Z}}$	A posterior probability distribution of the robot pose in the presence of observations
$\mathbb{R}^d$	A d-dimensional Cartesian space of real numbers
ρ	Size of the learning epoch
σ	Spread of the Gaussian filter; or it is the neighborhood radius around a winner neuron; or a scoring factor to instantiate a new neuron in IGBN
σ_0	Initial neighborhood radius around winner neuron
σ_u	Horizontal spread of the Gaussian kernel in frequency domain

σ_v	Vertical spread of the Gaussian kernel in frequency domain
$\mathbf{\Sigma}_t$	A matrix describing the covariance in robot's pose and landmarks' positions at current time t
$\mathbf{\Sigma}_{k,m}^{(n)}$	A matrix describing the covariance in the m^{th} landmark's position for the n^{th} particle at an arbitrary time instant k
τ	Current iteration number; or error tolerance factor
θ	Heading direction of the robot in a two-dimensional Cartesian space
$\mathcal{U}_t$	A set of all the motion commands issued to the robot up to time instant t
$\varepsilon_{a,b,c}$	Excitatory weight mask for a 3D Gaussian
$\varepsilon_i^{(k)}$	Quantization error due to neuron i at a time instant k
$\varepsilon_r^{(t)}$	SSE in the newly instantiated neuron
$\varepsilon^{(t)}$	A vector of quantization error in the network up to present time
φ	Constant for global inhibition in the RatSLAM system
$\varphi(i,j)$	Gaussian RBF between two arbitrary inputs indexed by i and j
$\mathcal{W}$	A random variable than instantiates to a neuron's index
$\mathcal{X}_t$	A set of all the pose configurations of the robot up to time instant t
$\mathcal{X}_t^*$	A sequence of optimal (least error) pose configurations of the robot up to present time t
$\mathcal{X}_k^{(n)}$	A sequence of pose configurations acquired by the robot for the n^{th} particle at a time instant k
$\mathcal{Z}_t$	A set of landmarks $\mathbf{z}_t^{(i)} : \forall i \in \{1, \ldots, N_o\}$ observed at a time instant t

List of Figures

List of Tables