FRANKFURT UNIVERSITY OF APPLIED SCIENCES

DOCTORAL THESIS

Visual Localization and Mapping in Seasonally Changing Outdoor Environments

---

Author:

M.Eng. Muhammad Haris

Supervisor:

Prof. Dr. Ute Bauer-Wersing, Frankfurt UAS

Reviewers:

Prof. Dr. Alexander Gepperth, Fulda UAS

Prof. Dr. Helge Ritter, Bielefeld University

A thesis submitted to Promotionszentrum Angewandte Informatik in fulfillment of the requirements for the degree of Doctor rerum naturalium (Dr. rer. nat.)

Submission Date: 03.02.2023

Disputation Date: 17.05.2023

Place and year of publication: Frankfurt am Main, 2023

Declaration of Authorship

I, M.Eng. Muhammad Haris, declare that this thesis titled, “Visual Localization and Mapping in Seasonally Changing Outdoor Environments” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly during candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. Except for such quotations, this thesis is my own work.
- I have acknowledged all primary sources of help.
- Where the thesis is based on work done by myself jointly with others, I have clarified what others did and contributed myself.

Sign:

Date:

Abstract

Visual localization and mapping refer to locating an agent in a scene and creating a representation of its surroundings using a camera as the primary source of perception. Localization and mapping are the fundamental prerequisites for autonomous robots, self-driving cars, and augmented reality applications. Despite decades of research and development in this domain, even state-of-the-art vision-based approaches struggle to perform in challenging outdoor conditions (e.g., lighting, structural, weather, and seasonal changes). Hence, this remains an active area of research and is of paramount importance for achieving autonomy over long periods. In addition to changing conditions, run-time and hardware constraints are crucial for practical applications. State-of-the-art methods often rely on 3D scene reconstruction for localization. However, 3D scene reconstruction is resource-intensive in terms of hardware requirements and computation time. Therefore, running it on a robot equipped with low-cost hardware is infeasible. The work presented in this thesis addresses these challenges, thus providing a robust, low-cost solution for mapping and localization in outdoor environments.

This thesis approaches the problem using a bio-inspired model based on unsupervised Slow Feature Analysis (SFA). The model reproduces the firing characteristics of Place and Head-Directions Cells found in the rodent's hippocampus. Recently, this model has been successfully used for outdoor localization. However, it is short-term stable w.r.t environmental changes, which limits its use over a long time. Moreover, it does not scale to large-scale environments, which limits its applicability to small settings. This work overcomes the first limitation by restructuring the long-term data to change the input data's perceived statistics. The restructuring allows the model to learn invariance to long-term scene changes. For large-scale localization, the proposed approach uses landmarks in a scene and learns to localize relative to them. The work also compares and analyzes the SFA-based approach with state-of-the-art methods w.r.t to localization accuracy, run-time, and hardware requirements. Successful localization and mapping enable many downstream tasks, like goal-directed navigation. The spatial smoothness of learned representation using SFA allows the approach to perform navigation using straightforward gradient descent without explicit path planning. All the experiments presented in this work were performed using real-world robot recordings, which enforces the feasibility of using the proposed method for practical applications.

Zusammenfassung

Der Prozess, die eigene Position im Raum aus Bildern zu bestimmen und eine Repräsentation der Umgebung aufzubauen wird als „Visuelle Lokalisierung und Kartierung“ bezeichnet. Dieser Prozess ist eine fundamentale Voraussetzung für die Funktion von autonomen Robotern, selbstfahrenden Fahrzeugen und Augmented Reality Anwendungen. Auch nach mehreren Jahrzenten Forschung und Entwicklung in diesem Gebiet stellt Visuelle Lokalisierung noch immer eine Herausforderung für den gegenwärtigen Stand der Technik dar, insbesondere unter Outdoor-Bedingungen mit wechselnder Beleuchtung, dynamischen Objekten, Wetterveränderungen und fortschreitenden Jahreszeiten. Diese Probleme gelten verschärft für Systeme mit Langzeit-Autonomie. Zu den genannten Herausforderungen durch die Umgebung kommen für die praktische Anwendbarkeit noch Weitere, wie Begrenzungen der Laufzeit und der verfügbaren Hardware. Gegenwärtige Methoden basieren oft auf 3D-Rekonstruktionen der Umgebung, deren Erstellung allerdings selbst auf moderner Hardware noch hohe Laufzeiten und Energieverbräuche bedingt. Ein autonomer Roboter mit energieeffizienter und kostengünstiger Hardware ist mit diesen Ansätzen auf absehbare Zeit unrealistisch. Die Beiträge in dieser Arbeit widmen sich den genannten Herausforderungen und erlauben damit die Realisierung eines robusten und kostengünstigen Systems zur Visuellen Lokalisierung und Kartierung in Outdoor-Umgebungen.
Diese Arbeit basiert auf Slow Feature Analysis (SFA), einem aus der Biologie inspirierten Ansatz des unüberwachten Lernens, in dem die zeitliche Statistik der wahrgenommenen Umgebung als Lernsignal genutzt wird. Dieses Modell wurde in der Vergangenheit genutzt, um das Verhalten von Ortszellen und Kopfrichtungszellen im Hippocampus von Nagetieren zu reproduzieren. Spätere Arbeiten belegen die grundsätzliche Nutzbarkeit für Lokalisierung auf Robotern in Outdoor-Umgebungen über Zeiträume von Minuten bis Stunden . Die Nutzung dieses Ansatzes für langzeitstabile Lokalisierung war bisher nicht gelungen. Darüber hinaus skalierte der bisherige Ansatz nicht auf größere Umgebungen. Diese Arbeit überwindet die erste Begrenzung durch einen Ansatz zur Restrukturierung der Zeitstruktur der visuellen Trainingsdaten und ermöglicht damit invariante Lokalisierung trotz Langzeit-Umgebungsänderungen. Die bisherige Größenbegrenzung der Umgebung wird hier durch die Einführung der Nutzung von Landmarken aufgehoben, anhand derer ein Roboter sich

lokalisieren kann. Diese Arbeit analysiert den neu entwickelten Ansatz und vergleicht Lokalisierungsgenauigkeit (accuracy), Laufzeit und Hardwareanforderungen mit dem gegenwärtigen Stand der Technik. Erfolgreiche Lokalisierung bildet die Basis für nachgeordnete Funktionen wie zielgerichtete Navigation. Die mit SFA gelernten räumlichen Repräsentationen zeichnen sich durch ihre Glattheit aus, die Navigation ohne explizite Pfadplanung durch einfachen Gradientenabstieg auf der Repräsentation erlaubt. Die Praxistauglichkeit aller Ansätze in dieser Arbeit ist durch die Anwendung auf einem realen Roboter und meist unter Outdoor-Bedingungen belegt.

Acknowledgements

I want to thank my supervisors, Ute-Bauer Wersing and Mathias Franzius, for putting their trust in me and offering an opportunity to pursue a Ph.D. I am short of words to describe their empathy towards me throughout these years. I highly commend their leadership skills in guiding me during the Ph.D. and providing me with enough freedom to do the research work. I also want to thank them for helping me with organizational things, proofreading, and many on- and off-topic discussions. I also want to thank my project advisor Joerg Deigmoeller for his valuable insights into visual localization and mapping, and Heiko Wersing for proofreading my research papers several times. Thanks to my master's thesis students, Kaushik Karanam and Matih Ullah, for their valuable contributions. Big thanks to Honda Research Institute GmbH for funding my Ph.D. position and all the colleagues there that made this time pleasant and memorable. Last but not least, I want to thank my wonderful family for their support and wishes during my Ph.D.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Autonomous navigation is the ability of an agent (e.g., a robot or a car) to plan its path towards a target location in an environment without human intervention. Consider a mobile robot (fig. 1.1) that wants to navigate to a goal location, for instance, a charging station. There are two fundamental requirements to perform this task successfully:

1. **Localization** (Where am I?)
2. **Mapping** (What does the world look like?)

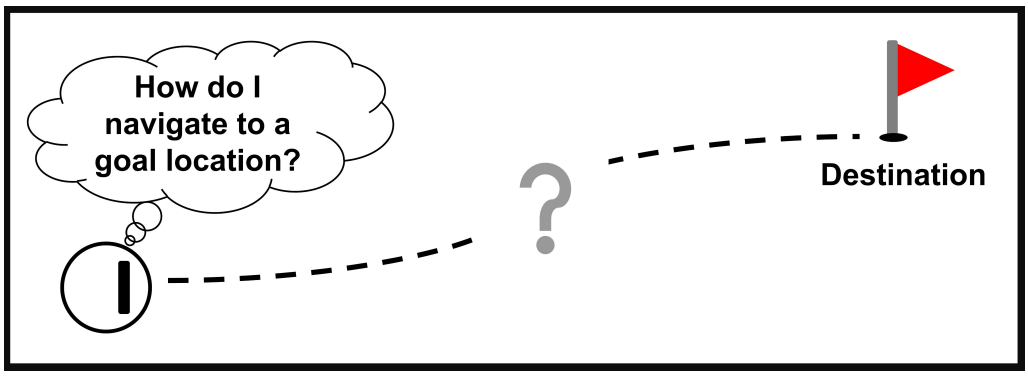


FIGURE 1.1: **Autonomous Navigation**: For successfully planning a path towards a goal location, an autonomous robot must be able to locate itself (localization) within a map (an internal representation of the robot's surroundings) of the environment.

Localization estimates an agent's current pose (x,y,ϕ) in an environment, while mapping creates a consistent representation of an agent's surroundings. Both technologies are prerequisites to implementing intelligent behavior. There is a plethora of applications that require localizing an agent in an environment, for instance: industrial robots, domestic robots (lawnmowers, vacuum cleaners), surveillance robots, self-driving cars, and more-recent delivery robots. These robots offer massive assistance

to humans in performing mundane or repetitive tasks. Various sensors and techniques allow localizing an agent in an environment. The following section presents an overview of such technologies.

1.1 Sensors for Localization

Wheel Odometry: The most straightforward and widely utilized approach for the pose estimation of an agent is wheel odometry. Odometry integrates an agent's motion based on wheel encoders. However, the integration accumulates errors and pose estimates drift strongly over time. In outdoor scenarios, wheel odometry can be strongly impacted by wheel slip or the accumulation of material on the wheels. Nevertheless, odometry is very accurate in the short-term (seconds to minutes) and can be integrated with other localization methods.

Inertial Measurement Unit (IMU): IMU uses an accelerometer and gyroscope to measure acceleration and angular velocity. Together, these sensors allow 6D pose (3D position and 3D orientation) estimation of an agent relative to the start location. Like wheel odometry, its estimate also drifts over time due to the accumulation of errors. Thus, it is unsuitable for long-term positioning estimation. Instead, it is typically utilized for short-term motion and combined with other localization methods for long-term operation.

Global Positioning System (GPS): GPS is an established localization system for outdoor use with line-of-sight between the receiver and at least four satellites. A GPS can localize anywhere with good satellite reception and allows long-term stable localization without manual intervention. Most available GPS systems use dedicated hardware and thus require little power and low computational effort. Accuracy with standard low-cost receivers is 1-2 meters but drastically reduces when occlusions between receiver and satellites occur (e.g., buildings, trees). It is not usable in some conditions, like indoors or tunnels, and signal reflections can also cause significant errors, especially on metal objects. Despite its limitations, it can be a cost-effective secondary localization source for mobile agents operating in outdoor environments.

LiDAR (Light Detection and Ranging): LiDAR uses an array of light pulses to measure the distance to nearby objects and yields a 3D point cloud of an environment. Distance measurements with laser sensors are highly accurate (often in the mm range) and can lead to precise localization. Moreover, it is less affected by appearance changes in the environment than vision-based approaches (except for seasonal

vegetation changes). However, LiDARs are unrealistically large and highly costly, which makes them less attractive to be deployed on a domestic robot (e.g., a lawnmower). Like vision-based approaches, these sensors can suffer from certain lighting conditions (e.g., low sun) and weather conditions (e.g., fog or rain).

Optical Cameras: Cameras can be employed to overcome most of the drawbacks of other sensors. For instance, camera-based methods are not affected by wheel slippage. In contrast to GPS, cameras can work in GPS-denied environments, like indoors. Moreover, compared to expensive LiDARs and differential GPS systems, vision-based approaches can work with a consumer-grade camera. These attributes make a camera-based solution appealing for localizing an agent in an environment. Moreover, cameras are versatile sensors that can also be used for other tasks besides localization, for instance, reading text from signs and interpreting colors.

1.2 Application Area

There are numerous applications of localization and mapping technology, e.g., from the phone in hand to the robots operating in space. However, the particular emphasis in this work will be on application scenarios for service robots (e.g., lawnmowers) in a limited environment like an apartment or a garden. The current capabilities of such robots are limited, i.e., they often rely on pre-installed wire guidance technology as a navigation aid. The movement patterns are also constrained to random line segments (fig. 1.2a) combined with some collision avoidance mechanism. For an efficient navigation strategy (fig. 1.2b), such robots should be able to create an internal representation of their surroundings (mapping) and locate themselves (localization) within the constructed map of the environment.

1.3 Visual Localization and Mapping

Visual Localization and Mapping enable a robot to build an environment map and estimate its location using a *camera* as the only source of perception. For localization and mapping, one can use multiple cameras (e.g., a stereo camera). However, this thesis addresses the problem using only a single camera (i.e., monocular visual localization and mapping). Despite its apparent advantages, the camera-based solution also has potential downsides, for example:

- Visual localization in constrained indoor environments can be considered a solved problem. However, in outdoor environments, it remains a challenging

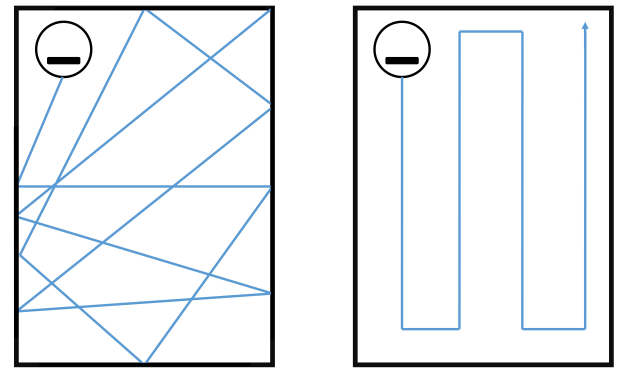


FIGURE 1.2: **Navigation Strategies for Domestic Robots**: (a) Domestic robots with a straightforward approach navigate randomly in an environment, i.e., typically going straight until they hit something or detect a pre-installed border wire. Afterwards, they turn in a random direction and start moving straight again. However, this is not optimal behavior since a robot could traverse unnecessarily in an already visited location of an environment. (b) On the other hand, for a more systematic navigation behavior, a robot must locate itself in an environment and maintain a map of its surroundings. In this way, these robots can plan optimal paths to efficiently accomplish a task (i.e., cleaning or mowing).

problem and, thus, an active research area (Toft et al., 2022). Many factors make visual localization difficult in outdoor environments, e.g., uneven terrains, lighting conditions, shadows, dynamic changes, weather, seasonal shifts, or structural changes. A localization method should ideally cope with such environmental changes, allowing long-term robust operation.

- Another limitation of a camera-based approach is image analysis, which is typically a computationally expensive task. Unlike a LiDAR sensor that provides a precise distance and location of a nearby object, a camera gives a raw image of a scene. Therefore, appropriate image-based algorithms are necessary to process the images that often require more computational power. Hence, an ideal localization method should be realizable on low-cost embedded hardware, especially for consumer household robots (e.g., lawnmowers and vacuum cleaners).

As discussed, each sensor has its limitation. A viable approach to overcome this issue is to combine multiple sensors (e.g., cameras, GPS, IMU, and odometry) and take their advantages while limiting the drawbacks of a particular sensor. However, sensor fusion is beyond the scope of this thesis, and the focus here is only on tackling the challenges posed by cameras. In addition to other aspects of visual localization

methods, the following are considered especially relevant here:

Temporal Generalization: The primary scientific target of this work is to advance vision-based autonomous localization towards substantial temporal environment variation (e.g., seasonal changes) and develop efficient learning mechanisms that can deal with this challenging requirement. Therefore, the aim is to concentrate only on algorithms that do not require prior knowledge of the environment.

Computational Requirements: Another critical aspect of a localization method is its feasibility of running on low-cost hardware, especially for real-world commercial applications. Therefore the aim is to develop a computationally efficient solution that can be implemented on embedded hardware without needing specialized hardware (e.g., a GPU).

Localization Accuracy: The target is to achieve better or similar absolute localization accuracy (in meters) compared to the state-of-the-art visual localization approaches.

Spatial Generalization: It is the ability of a localization approach that allows a robot to locate itself in previously unvisited areas in an environment. For example, in the context of learning-based methods, this means how well the learned spatial representation generalizes to new positions (unseen during training) in the same environment. It is an essential aspect of a localization method that shows its strength.

1.4 Thesis Outline

This thesis employs a biologically-motivated model to perform visual localization and mapping, as many animals show excellent localization and navigation abilities in natural environments. In 1971 O'Keefe and Dostrovsky found that the firing activity of Place Cells is highly related to the rat's position in an environment (O'Keefe et al., 1971). In (Taube et al., 1990), Head-Direction Cells have also been recognized later to play an essential role in rodent localization ability. Both Cell types strongly depend on visual input (Jeffery et al., 1999). Earlier learning approaches (Franzius et al., 2007) show that a hierarchical model can reproduce the activity of Place or Head-Direction Cells if trained from the visual cues of a virtual rat in an unsupervised way. The model uses the concept of Slow Features Analysis (SFA) (Wiskott et al., 2002), and the intuition behind it is that behaviorally meaningful information (e.g., position or orientation of an animal in space) changes on a slower timescale

compared to the primary sensory input (e.g., pixel values in a video). The learned latent representations are well-suited for localization and navigation. The later chapters of this thesis will present more details on the approach.

Visual localization and mapping are well-researched areas. More than two decades of research exist in this domain, ranging from classical vision-based to more recent learning-based methods. Chapter 2 will present an overview of such approaches that aim to perform localization and mapping using a camera as the only exteroceptive sensor.

Chapter 3 is dedicated to introducing the unsupervised Slow Feature Analysis (SFA) algorithm, the primary method for learning localization-relevant latent representation in this work. The chapter first presents the high-level intuition of the approach before diving into the mathematical formulation of the problem and its implementation details. Due to the high input dimensionality of visual images, SFA can not be applied directly to the visual data. Therefore, the chapter also introduces a feed-forward converging hierarchical SFA network to process such data. Later, some example applications of the approach are presented, along with the idea of how SFA can help extract a latent representation suitable for localization and navigation.

Chapter 4 explains the procedure of generating and capturing the visual data. The experiments were performed in a perfectly-controlled simulated environment, an indoor area, and two garden-like outdoor environments. Further, it presents the procedure to acquire ground-truth metric coordinates (x, y) for real-world data, which serves as a reference to evaluate the accuracy of the localization methods.

Localization and mapping methods typically follow a modular pipeline to construct a scene representation and estimate an agent's location from input data. Chapter 5 introduces the proposed visual mapping and localization pipeline in this work. The mapping pipeline uses the input image stream to learn a scene representation with SFA. The chapter further describes different pre-processing techniques and details regarding the SFA network architecture. The last section presents the localization pipeline that computes the SFA output of an input image using the trained model. For evaluation purposes, the procedure to map the learned SFA representation to the corresponding metric coordinates (x, y) is also described.

In outdoor scenarios, changing conditions (e.g., seasonal, weather, and lighting effects) have a substantial impact on the appearance of a scene, which often prevents successful visual localization. Chapter 6 introduces the proposed approach solely based on SFA to tackle this challenging problem. The method uses SFA's invariance learning capabilities by restructuring the temporal order of the training data. This restructuring scheme allows SFA to learn invariance to the undesired variables (here:

conditions) and encode localization-relevant representation for long-term robust operation.

Chapter 7 introduces an alternative method to perform localization that identifies landmarks in a scene and then learns localization relative to them using SFA. The approach performs well in a large-scale outdoor environment. However, this relies on hand-labeled data to train a CNN to recognize landmarks, which is a tedious task. Thus, the chapter also presents a novel approach that allows a robot to learn landmarks for localization with a human cooperatively. This approach uses pre-trained detectors of everyday objects to learn new landmarks in a scene, requiring minimal human supervision. Hence, the method bootstraps the landmark learning process and removes the need to manually label large amounts of data.

State-of-the-art methods use the 3D structure of a scene for precise visual localization. However, 3D scene reconstruction is resource intensive in terms of hardware requirements and computation time, making it infeasible to run on low-cost embedded hardware. Unsupervised spatial representation learning with Slow Feature Analysis (SFA) enables computationally inexpensive localization and mapping. Chapter 8 analyzes SFA-based and state-of-the-art methods in two distinct settings: short-term temporal and extreme spatial generalization, and compares the methods based on localization accuracy and computation time.

SFA enables a mobile robot to learn a spatial representation of its environment from images captured during an exploration phase. After the unsupervised learning phase, a subset of the resulting representations code for the robot's position. The representation is spatially smooth and implicitly encodes the average travel time during exploration. Following the SFA gradient allows the robot to navigate even around obstacles without any planning. Earlier work (Metka et al., 2017) showed this basic principle in noise-free simulation using two virtual cameras on a robot. Chapter 9 presents an extension to the approach to make it more robust and computationally efficient. The experiments were performed on a lawn mower robot with a single camera for navigation in free space and around obstacles.

Chapter 10 wraps up the thesis, provides the main takeaways and highlights the scientific and technological impact of the work.

1.5 Accepted Publications

The following are the accepted publications in the context of this thesis:

1. Haris, M., Metka, B., Franzius, M., & Bauer-Wersing, U. (2017). Condition invariant visual localization using slow feature analysis. New Challenges in Neural Computation (NC2). Machine Learning Reports, Frank-Michael Schleif.

2. Haris, M., Franzius, M., & Bauer-Wersing, U. (2018). Robot navigation on slow feature gradients. In International Conference on Neural Information Processing (ICONIP, pp. 143-154). Springer, Cham.

3. Haris, M., Franzius, M., & Bauer-Wersing, U. (2019). Robust outdoor self-localization in changing environments. In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS, pp. 714-719). IEEE.

4. Franzius, M., Metka, B., Haris, M., & Bauer-Wersing, U. (2020). Unsupervised Learning of Metric Representations with Slow Features from Omnidirectional Views. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW, pp. 638-639). IEEE.

5. Haris, M., Franzius, M., Karanam, K. S. K., & Bauer-Wersing, U (2020). Visual Localization and Mapping with Hybrid SFA. Conference on Robot Learning (CORL), 2020. Proceedings of Machine Learning Research.

6. Haris, M., Franzius, M., & Bauer-Wersing, U. (2021). Unsupervised Fast Visual Localization and Mapping with Slow Features. In IEEE International Conference on Image Processing (ICIP, pp. 519-523). IEEE

7. Haris, M., Franzius, M., & Bauer-Wersing, U. (2022). Learning Visual Landmarks for Localization with Minimal Supervision. 21st International Conference on Image Analysis and Processing (ICIAP, pp. 773-786).

8. Haris, M., Franzius, M., & Bauer-Wersing, U. (2022). Physical Interactive Localization Learning. In IEEE International Conference on Advanced Robotics and its Social Impacts (ARSO). IEEE

Chapter 2

Literature Overview

Visual mapping creates a scene representation (i.e., a map of the environment) using a camera, and localization infers the camera's (or robot's) location on the map using a single snapshot (i.e., a single image or a set of stereo images). Localization, in general, is a prerequisite for many robotic applications, autonomous cars, and AR/VR systems.

Most approaches use a pre-built map or build such a map online. However, mapless systems exist, e.g., motion integration systems based on feature tracking or optical flow. These systems accumulate localization errors over time, like non-visual odometry. Image retrieval (Arandjelovic et al., 2014; Balntas et al., 2018) is a method for finding the best match between a query image and an image database, where the image database can be position annotated (e.g., images with GPS position). Standard image retrieval methods thus return the position of the best matching database image, but interpolation between matches is also possible. Due to pose approximation, such methods can only enable coarse localization compared to highly-precise map-based methods (Sattler et al., 2019). The rest of this chapter will focus on methods that employ a map for performing visual localization.

Figure 2.1 shows a diagram of different map-based visual localization approaches. These can be divided into simultaneous (i.e., online) and non-simultaneous (i.e., offline) visual localization and mapping techniques. As the name indicates, simultaneous methods (e.g., vSLAM) create an environment map and perform localization within the map concurrently. In contrast, non-simultaneous approaches (e.g., learning-based) have separate mapping and localization phases; thus, these approaches rely on a pre-built environment map for localization. The pre-built map can have an explicit (e.g., 3D point cloud reconstruction) or implicit (e.g., neural network weights) representation.

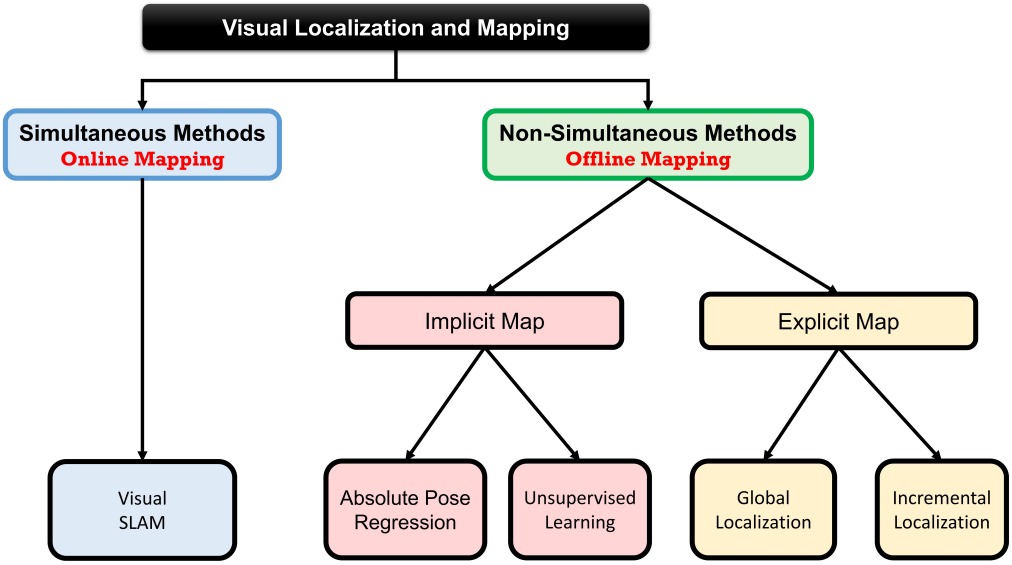


FIGURE 2.1: **Localization and Mapping Overview**: Simultaneous approaches perform mapping and localization in an online manner. In contrast, non-simultaneous methods create an environment map (i.e., explicit or implicit map representation) in an offline step and perform localization afterwards.

2.1 Structure-based Methods

The most traditional approach for visual localization relies on identifying point-like features in images. It first builds a 3D map of feature positions in space and then estimates the camera's 6D pose relative to such features (Sattler et al., 2011; Sattler et al., 2017; Liu et al., 2017; Sarlin et al., 2019). The process of creating a 3D scene reconstruction (i.e., a map) using images from different locations and viewpoints of the same scene is known as structure-from-motion (SfM) (Moulon et al., 2013; Schönberger et al., 2016). The localization procedure consists of extracting and matching 2D features of a test image to a 3D point cloud reconstruction computed from a structure-from-motion (SfM) step. Afterwards, a subsequent step computes a test image's pose by solving a Perspective-n-Point (PnP) problem (Kneip et al., 2011), typically using RANSAC (Chum et al., 2008). Various feature descriptors have been proposed in the past (e.g., SIFT, SURF, BRIEF) to extract point-like features from the visual scene, where the features are preferably unique and invariant to changes in perspective and lighting. However, these handcrafted features are often not unique and vary strongly with camera perspective and environmental conditions. Thus the methods that rely on such features offer limited robustness against temporal changes. To overcome this issue, several data-driven approaches are also proposed that employ deep architectures (Schönberger et al., 2017) to learn point-like features (DeTone et al., 2017; Dusmanu et al., 2019). Although structure-based methods offer

precise localization accuracy in most scenarios, their performance in adverse conditions is still limited (Toft et al., 2022). Another limitation of such approaches is that both the mapping and localization steps are typically very costly in computation and memory. Several authors (Sarlin et al., 2019; Humenberger et al., 2020) incorporate image retrieval methods in the structure-based localization pipeline. These methods first retrieve the most relevant images and then look for correspondences within those images. Thus, it helps to reduce the search space and speed up the algorithm execution time. However, such methods then heavily rely on the quality of image retrieval methods. To summarize, despite being the most precise localization methods, changing conditions and computationally expensive algorithms are still the main challenges these methods suffer.

2.2 Learning-based Methods

In contrast to structure-based approaches, learning-based methods apply machine learning techniques to perform regression from images to a 2D, 3D, or 6D camera pose. These approaches are typically instantaneous, i.e., localization is performed on each image independently. Unlike structure-based approaches, learning approaches do not require calibrated cameras because pose regression is learned from the statistics of scene appearance in an end-to-end manner with minimal assumptions. It is, however, important that the camera setup stays the same between training and testing (e.g., change of focus, zoom, camera chip, or lens) unless some form of image augmentation is used. Like structure-based methods, localization learning methods require an initial training phase for each new environment. Typically, supervised learning methods generalize poorly in space (i.e., extrapolation and interpolation of training poses), whereas generalization over environmental conditions (e.g., lighting and weather) can be achieved if such data is part of the training set. State-of-the-art methods for localization learning have an implicit map of the environment, i.e., no explicit 2D or 3D model can be displayed to a human, e.g., for programming tasks or indicating locations of objects. In contrast to structure-based methods, end-to-end learning methods have minimal assumptions on the environment's statistics, which means that these methods do not require hand-engineered visual features. Thus, these methods can be trained to work in unusual settings and situations (e.g., night-time, desert-like structures, or repeating textures) where structure-based methods fail or require manual intervention.

In recent times, the training of convolution neural networks (CNNs) for visual localization has gained significant interest. Since many localization pipelines are highly modular, several researchers have attempted to learn only parts of them using CNNs

(Meng et al., 2017; Schönberger et al., 2018). Others use features from pre-trained convolutional neural networks for the task of place recognition (Sünderhauf et al., 2015; Sünderhauf et al., 2015). Features obtained from different layers are invariant w.r.t viewpoint and condition changes. Absolute Pose Regression (APR) describes a class of approaches for supervised end-to-end learning of absolute metric poses from images. In (Kendall et al., 2015), authors have trained a convolutional neural network in an end-to-end way that learns to regress the 6D pose from a single monocular image. The learned features are robust in challenging scenarios, for instance, varying lighting conditions and weather effects. As an extension, the authors have fine-tuned the model using a geometric loss function (Kendall et al., 2017) and achieved a higher localization accuracy than the base model. Several other architectures have been proposed to improve the localization accuracy of such methods (Brachmann et al., 2017; Valada et al., 2018). Despite impressive results, end-to-end learning for visual localization does not compete with the structure-based methods in terms of pose accuracy. Recent studies (Sattler et al., 2019) have shown that CNN approaches do not necessarily generalize well beyond the training data in practical situations, limiting their applicability in the real world. The work highlights that end-to-end learning is currently competing with image retrieval methods rather than pose estimation using 3D geometry. Moreover, supervised pose regression has significant training overhead for each new environment in terms of computational load and is practically infeasible without a GPU. Supervised learning, especially on dedicated hardware like GPUs, has drastically improved in recent years. Still, even with such hardware, training time ranges from hours to days. Additionally, ground truth poses are typically generated by structure-based methods before training time, adding the overhead for map generation to the process. The training process is infeasible on embedded devices unless they are supported by dedicated hardware like NVIDIA Jetson[1]. However, unlike structure-based methods, APR-based methods require less memory and have a constant inference time after training. To summarize, these methods can learn invariance to environmental conditions and thus can perform better than structure-based approaches in such scenarios. However, the potential limitations include costly labeled-data generation (typically with SfM), computationally expensive mapping phase, high-end hardware requirements (i.e., GPUs), and limited generalization capabilities.

In contrast to end-to-end supervised learning, semi-supervised and unsupervised localization learning strongly reduces or eliminates the need to generate costly ground

[1] https://www.nvidia.com/de-de/autonomous-machines/embedded-systems/

truth labels. In these approaches, a relatively low-dimensional environmental representation is generated without labels, which is then used for localization or navigation tasks. This work is based on an unsupervised Slow Feature Analysis method (Wiskott et al., 2002), which will be described in greater depth in chapter 3.

2.3 Simultaneous Localization and Mapping (SLAM)

SLAM creates a map of an unknown environment and locates an agent within this map concurrently. Thus, in contrast to structure- and learning-based methods, SLAM is an online method, i.e., localization is available from the beginning and without a dedicated training phase. SLAM using cameras only is also referred to as visual SLAM (i.e.,vSLAM). The focus in this section will be on approaches that use vision as the only input modality. However, non-visual sensors can also be employed for SLAM algorithms (Durrant-Whyte et al., 2006; Bailey et al., 2006). Without a map, SLAM reduces to odometry, which accumulates drift over time. Visual SLAM approaches can be divided into indirect and direct methods. The indirect approach detects and extracts geometric features (also known as key points) from images and matches them across consecutive frames to recover the camera pose. In contrast, the direct approach considers the entire image and works on pixel intensity without preprocessing. Both methods have their pros and cons. For instance, indirect techniques are much faster due to the usage of sparser key points and, thus, are highly relevant for real-time applications. On the other hand, the direct approaches are much more accurate and can perform better in environments with sparse texture (Zhu et al., 2021). The following paragraphs will highlight some of the works in both categories. Indirect methods are all about detecting, extracting, and matching geometric features across images to estimate pose and create a sparse scene representation. Different types of geometric features have been used by different researchers, e.g., points (Rublee et al., 2011; Bay et al., 2008), corners (Shi et al., 1994), and lines (Gioi et al., 2010). The research community predominately uses methods based on point-like features due to their simplicity and applicability in real-world applications. However, several approaches (Pumarola et al., 2017; Gomez-Ojeda et al., 2019) have also been proposed that combine different geometric features (i.e., point, line, plane) for improving the localization accuracy and robustness of the SLAM-based algorithms. MonoSLAM (Davison et al., 2007) is the first-ever single camera-based algorithm that achieves real-time performance for simultaneous mapping and localization. The front end uses shi-Tomasi features for the detection and matching steps, while the back end is based on the Extended Kalman Filter (EKF) (Smith et al., 1990) that optimizes and generates a sparse environment map. Despite its significance in the

SLAM community, the method does not apply to larger scenes due to its computational complexity. PTAM (Parallel Tracking and Mapping) algorithm (Klein et al., 2007) splits the tracking and mapping into separate threads on a CPU for tackling the computational issue of MonoSLAM. The method ensures real-time performance; however, the absence of a loop closure detection module causes the system to accumulate errors over a more extended period. In SLAM terminology, a loop closure occurs when mapped features (i.e., landmarks) are re-detected from a new position, which facilitates in reducing both landmark errors in the map and pose errors of the agent. ORB-SLAM (Mur-Artal et al., 2015) is considered a successor of the PTAM algorithm and is also one of the most widely used visual SLAM methods. Unlike PTAM, the authors added a vision-based loop detection module based on the bag of words model (Galvez-López et al., 2012) to prevent the system from accumulating errors over time. As the name indicates, the method uses ORB features for detection and matching steps. In contrast to PTAM, it runs three threads simultaneously (i.e., tracking, mapping, and loop closing). The method is well known for its performance on CPU in real-time and also has been used in applications ranging from hand-held devices to self-driving cars. Since its first version, several improvements to the method have been proposed (i.e., ORB-SLAM2 (Mur-Artal et al., 2017) and ORB-SLAM3 (Campos et al., 2021)). The newest version allows the technique to perform re-localization even when the tracking is lost due to poor visual information and enables integration of multiple sensor modalities for SLAM (e.g., visual-inertial SLAM). Despite their superior performance, these methods are highly dependent on environmental features, which makes them less applicable in textureless environments.

In contrast, direct methods eliminate the need to perform detection and feature-matching steps by directly operating on the entire image. The advantage of retaining complete image information is that these methods can adapt to environments with sparser features. In contrast, the performance of feature-based techniques would deteriorate under such conditions. Unlike sparse scene mapping using indirect approaches, direct methods generate a semi-dense representation of the environment. However, it is also possible to construct dense maps using these methods. DTAM (Dense Tracking and Mapping) is the first practical method in the direct approaches for visual SLAM (Newcombe et al., 2011). The technique tracks camera motion by comparing the current image with the synthetic model view generated from the map. Due to dense map generation, the method offers more robust performance in featureless environments and can also handle motion blur. However, the main limiting factor is that real-time performance can only be achieved using a GPU. Large Scale Direct SLAM (LSD SLAM, Engel et al., 2014) is a well-known direct visual SLAM

method, which uses direct image alignment for camera tracking, minimizes the photometric error, and generates a semi-dense scene representation. The technique enables precise large-scale localization and works in real time on a CPU. Semi-dense Visual Odometry (SVO, Forster et al., 2014) extracts features from keyframes and uses the direct approach for pose estimation on the tracked features between consecutive camera frames. Its advantages include constant run time, higher framerate, and localization accuracy. However, it is not a complete SLAM system due to the absence of loop closure detection or a global map optimization module. In contrast to SVO, Direct Sparse Odometry (DSO, Engel et al., 2018) does not extract image features. Instead, it evenly subsamples image pixels across regions with a high gradient, making the approach effective for featureless environments. Although the method does not incorporate a re-localization module, the resulting maps have a relatively small drift compared to other visual odometry approaches.
To summarize, the algorithms within the visual SLAM category range from sparse indirect to dense direct methods. In general, decades of SLAM research exist, yet there is still not a single standout method that can work in any environment using any sensor type. Instead, the employed solution depends highly on the robot, environment type, and computational constraints (Cadena et al., 2016).

2.4 Semantics-based Methods

An exciting alternative to overcome some challenges posed by visual localization is to use semantic information like objects or landmarks. The breakthrough in Convolutional Neural Networks (CNNs) performance for object detection (Krizhevsky et al., 2012) has allowed researchers to incorporate it into the traditional SLAM pipeline, which has led to the creation of semantically meaningful maps. Earlier work (Bao et al., 2012) in this field has extended the structure-from-motion (SfM) pipeline for joint estimation of camera parameters, scene points, and object labels. However, its computational complexity limits the method's ability to operate in real time. SLAM++ (Salas-Moreno et al., 2013) detects a set of known object instances and maps them with an object pose graph. Other object-level SLAM methods (Frost et al., 2018; Sucar et al., 2018) use object detection in a scene to solve the problem of scale uncertainty and drift of monocular SLAM. In (Gálvez-López et al., 2016), the authors considered an algorithm based on bags of binary words (Galvez-López et al., 2012) that leverages a massive database of objects. To improve the map and find the real scale, the monocular SLAM algorithm and the object recognition algorithm exploit not only the object rigidity constraints but also the earlier observations acquired by SLAM that serve as cues for the location of the objects in the present image. This

approach leads to faster and more detections. Thus, it provides more geometrical constraints to SLAM. However, the method does not show its ability to work with dynamic objects. QuadricSLAM (Nicholson et al., 2018) is an object-oriented SLAM that does not rely on prior object models. Rather it represents the objects as quadrics, i.e., spheres and ellipsoids. It jointly estimates a 3D quadric surface for each object and camera position using 2D object detections from images. In (Bowman et al., 2017), the authors use semantic objects, e.g., chairs and doors, in a semantic SLAM approach. The proposed system performs continuous optimization over the poses while it discretely optimizes the semantic data association. Therefore, it divides the metric semantic SLAM into two sub-problems. The method couples the inertial, geometric, and semantic information into a single optimization framework. Fusion++ (Mccormac et al., 2018) is another object-level SLAM system that focuses on indoor scene understanding with an RGB-D camera. The system produces semantically labeled Truncated Signed Distance Function (TSDF) reconstructions of the objects with the Mask-RCNN object detector. Afterward, the system adds the TSDF object instances to the map for tracking, graph optimization, and re-localization. In (Hosseinzadeh et al., 2019), the authors include an object detector in a monocular SLAM framework to represent generic objects as landmarks. Although camera localization is precise with sparse point-based SLAM, it lacks semantic information. CNN-based object detectors perform well in providing essential information about the objects from single images. Therefore, this method uses CNN-based object and plane detectors to construct a sparse semantic map for localization. Semantic objects and plane structures, along with their completed point clouds, are included in the SLAM bundle adjustment. However, the method requires a post-processing step to recover the scale of the inserted objects, which is expensive and error-prone. In (Thrun, 1998; Zhao et al., 2018), the authors use machine learning-based approaches for localization in indoor environments relative to landmarks. CubeSLAM (Yang et al., 2019) combines 2D and 3D object detection with SLAM pose estimation by generating cuboid proposals from single-view detections and optimizing them with points and cameras using multi-view bundle adjustment. In (Parkhiya et al., 2018), the authors construct category-level models with CAD collections for real-time object-oriented monocular SLAM. The authors develop a rendering pipeline that helps in generating large amounts of datasets with limited hand-labeled data. The proposed system learns 2D features from category-specific objects, such as chairs and doors, with a deep network. The next step then matches the learned features to a 3D CAD model for pose estimation of the semantic object. The final step adds these semantic objects and the estimated robot's pose from VO to an optimizing graph framework for obtaining a metrically correct robot pose.

Most existing literature on object-SLAM considers indoor scenes or outdoor autonomous driving scenarios. In both cases, it is possible to directly use a pre-trained CNN to identify enough objects in a scene without training a detector on custom objects. However, the problem arises when a scene lacks pre-trained objects. In this case, training a custom object detector becomes a necessary pre-condition for most object-based localization approaches, which slows down the process of adapting such methods in new environments.

2.5 Bio-inspired Methods

In contrast to technical approaches, various methods for visual localization take inspiration from neurobiological systems. The primary motivation for such methods is that many animals exhibit excellent localization and navigation capabilities and quickly find their way to a food source or nest location, even in challenging environmental conditions. The paths taken by these animals may not be optimal, but they are flexible and rapidly generated. This motivated many researchers to model the animal's spatial cognitive processes and use them to propose lightweight and efficient navigation solutions for artificial agents (Milford et al., 2004; Cuperlier et al., 2007; Metka et al., 2018; Chen et al., 2019; Espada et al., 2019).
The discovery of the Place and Head-Direction Cells (O'Keefe et al., 1971; Taube et al., 1990) in the hippocampus region of the rodent brain has remarkable significance in this realm. Place Cells are responsible for encoding the animal's location in the environment, while Head-direction Cells encode the animal's orientation in space. In (Arleo et al., 2000), the authors present the computational model of these cells. The combination of visual cues and path integration in a Hebbian learning framework creates a population of Place Cells, enabling robot navigation. The work (Barrera et al., 2008) follows a similar approach and encodes unique places and their spatial relations in a topological map. The model also learned and unlearned navigation actions towards specific goal locations. Ants combine path integration and visual cues to navigate to their nest in desert environments (Wehner et al., 1996; Collett et al., 2013). Lambrinos et al., 2000 implement ant navigation on a real robot in a desert environment using artificial landmarks. The method uses wheel odometry for path integration and a compass system for obtaining global head direction. The compass allows the technique to align the current panoramic view to the stored snapshot of the target location. Afterwards, the computation of homing vector based on image matching enable navigation. The researchers (Philippides et al., 2011) suggest that ants might employ the skyline region to determine the homing direction due to their dichromatic vision with peak sensitivities in the ultraviolet and green range. In (Stone

et al., 2014), the work presents topological localization results using binary images that encode sky and non-sky pixels representing places along a 2km route. Another vision-based SLAM algorithm (Milford et al., 2008) derived from the computational model of a rodent's hippocampus builds the map online and performs re-localization in previously familiar scenes. The authors demonstrate successful mapping of a 66-kilometer urban road network with a single webcam using the proposed biologically inspired model. In (Milford et al., 2014), the authors link biology and robotics and provide an overview of conventional and biomimetic models for goal-directed navigation.

Metka et al., 2013 employed a bio-inspired model for visual localization on a real robot in an outdoor environment. The model strongly depends on the movement statistics of the robot during the learning phase. Thus, the learned representation encodes localization-relevant (i.e., position) information if the robot's orientation changes faster than its position during training. However, it is undesirable to mount a rotating camera on a robot due to mechanical stability. Thus, the authors artificially generated the robot rotation by using an omnidirectional mirror, which allows for generating an orientation-invariant representation of the robot's position. The work has proven the concept of SFA-based localization for outdoor scenarios with mean localization errors of less than 6%. To summarize, it establishes a system that learns instantaneous representations of the robot's position in an unsupervised learning process and achieves similar or equal performance compared to state-of-the-art visual SLAM methods, i.e., ORB and LSD-SLAM (Mur-Artal et al., 2015; Engel et al., 2014), in different test scenarios (Metka et al., 2018). Despite the impressive results, the learned representations are likely to be valid only for a shorter period, i.e., when the image statistics are similar to the training. Hence, this model has short-term stability w.r.t environmental conditions, which fail to perform localization if tested with data having different environmental conditions than the training set.

In summary, various image-based methods exist for mapping and localization, and these methods fall mainly into geometric feature-based and learning-based categories. Despite remarkable results, many challenges still exist which are crucial to address for achieving long-term robot autonomy. Therefore, this thesis addresses several open challenges (e.g., robustness to changing conditions and computationally feasible algorithms) and proposes solutions to advance vision-based localization and mapping that would foster long-term robust robot operation in the real world.

Chapter 3

Slow Feature Analysis

This chapter introduces an unsupervised learning approach called Slow Feature Analysis (SFA). It is the core algorithm that forms the basis of this work in the context of learning the spatial representation of a robot's surroundings. SFA, motivated from computational neuroscience, is based on slowness as a learning principle that extracts slowly varying features from a quickly changing input signal.

3.1 Background Motivation

We can characterize a visual scene into a high and low-level representation. A high-level representation encodes a scene's essential aspects like an object's identity and position. In contrast, a low-level representation encodes local features such as the color of the individual pixels. Thus, the former representation has a higher abstraction level since it explicitly contains the relevant information. One of the differences between these two representations is the time scale on which they vary, as the local features tend to change on a faster time scale compared to high-level features. For instance, the structure and presence of an object are likely to remain unchanged over a more extended period than its localized features (i.e., texture in a small patch), which are more sensitive to perspective or environmental changes. This observation forms the basis of the Slow Feature Analysis (SFA) algorithm, a successful extraction of slowly varying features from the quickly changing input signal can help to generate a useful representation of the visual environment.
Consider a movie strip of a visual scene as shown in fig. 3.1. Just by looking at the strip from left to right, we can easily recognize the movement pattern of the monkey that leaves the view towards the left. However, this high-level representation is distributed non-linearly over the response of sensory receptors in the retina, and the brain must reconstruct this information from their response to produce a meaningful representation. By extracting these slowly varying features from the input signal,

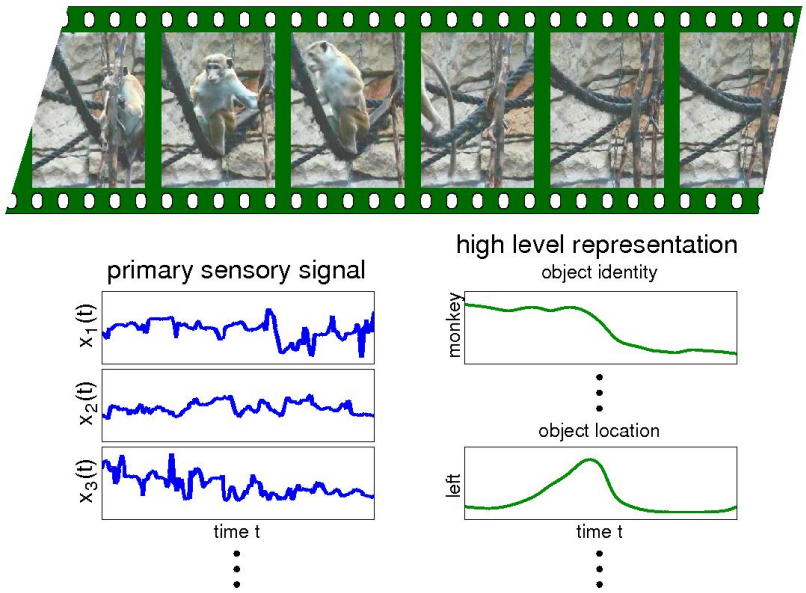


FIGURE 3.1: **Slowness Principle** - Reproduced from Wiskott et al., 2011: The movie strip in the figure represents a natural visual scene in which the monkey is leaving the field of view towards the left. Nevertheless, this high-level information about the monkey's presence and position (green traces) is non-linearly distributed over the response of sensory receptors (blue traces) in the retina. These high-level signals vary smoothly over time. The brain may use this insight to learn a representation that transforms the blue traces into green ones.

we can recover the underlying external causes of this signal. This is the objective of Slow Feature Analysis (SFA), i.e., to learn a representation that instantaneously maps a highly varying input signal to a slowly varying output signal.

3.2 Mathematical Definition

Slow Feature Analysis (Wiskott et al., 2002) transforms a multidimensional time series $\mathbf{x}(t)$, in our case images, along a trajectory, to slowly varying output signals. The objective is to find instantaneous scalar input-output functions $g_j(\mathbf{x})$ such that the output signals

$$y_j(t) := g_j(\mathbf{x}(t)) \tag{3.2.1}$$

minimize

$$\Delta(y_j) := \langle \dot{y}_j^2 \rangle_t \tag{3.2.2}$$

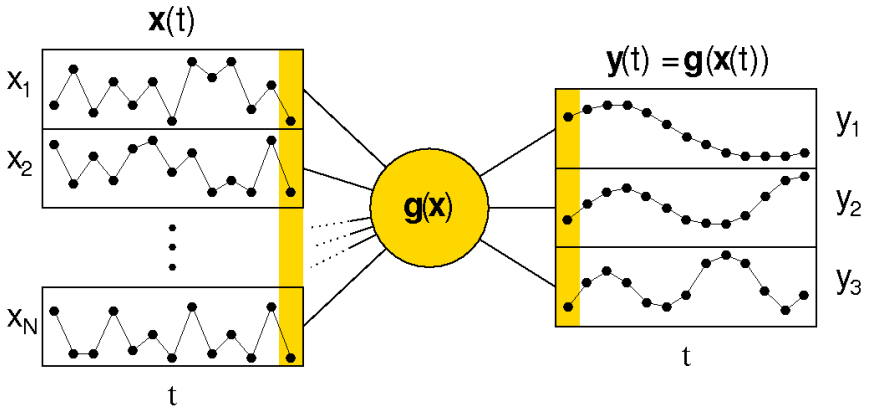


FIGURE 3.2: **Schematics of the optimization problem solved by SFA** - Reproduced from Wiskott et al., 2011: Given a high-dimensional time-series signal $\mathbf{x}(t)$, SFA learns instantaneous functions $\mathbf{g}(\mathbf{x})$ that transforms the input into slowly varying output signals, $\mathbf{y}(t)$. Functions $\mathbf{g}(\mathbf{x})$ must be instantaneous; for instance, one time slice of the output is based on one time slice of the input, as indicated by the yellow color.

under the constraints

$$\langle y_j \rangle_t = 0 \text{ (zero mean)}, \tag{3.2.3}$$

$$\langle y_j^2 \rangle_t = 1 \text{ (unit variance)}, \tag{3.2.4}$$

$$\forall i < j : \langle y_i y_j \rangle_t = 0 \text{ (decorrelation and order)} \tag{3.2.5}$$

with $\langle \cdot \rangle_t$ and $\dot{y}$ indicating temporal averaging and the derivative of y, respectively. The optimization problem solved by Slow Feature Analysis (SFA) is shown in fig. 3.2. The Δ-value in equation (3.2.2) defines the temporal variation of the output signal $y_j(t)$, and its minimization is the objective function. A lower value indicates less signal variation over time, thus means slowly varying signals. The constraints in equations (3.2.3) and (3.2.4) normalize all output signals to a standard scale to make their time derivative comparable and avoid the trivial solution, $y_j(t)$ = constant. The constraint in equation (3.2.5) enforces that different output signals code for different aspects of the input. It also induces an order such that output signals are ordered by the degree of their invariance. It is important to note that these transformations must be instantaneous, so any trivial solution like a low-pass filtering is not possible.

3.3 The Algorithm

The optimization problem, as defined in the previous section, is one of variational calculus, which is generally difficult to solve as it requires optimizing over functions, g_j, not over a set of parameters. However, if the input-output function components, g_j, are constrained to be a linear combination of a finite set of nonlinear functions

(e.g., all polynomials of degree two), one can transform the variational problem into a more conventional optimization problem. Below are the steps to find the function $\mathbf{g}(\mathbf{x})$ that extracts the slowly varying features from the input signal (Wiskott et al., 2011):

1. Expand the input signal into a space of nonlinear functions.
2. The expanded signal is normalized, so all constraints required by the SFA optimization problem are satisfied. It is referred to as whitening or sphering and can be done by applying principal component analysis to the data.
3. Temporal variation is measured in the normalized space by calculating the time derivative of the sphered signal.
4. The slowest varying directions are extracted by finding the direction of least variance of the time derivative signal. This step can also be performed with the principal component analysis in which the directions are the principal components with the smallest eigenvalues of the derivatives.

This work uses the MDP (Zito et al., 2009) implementation of SFA, which is based on solving a generalized eigenvalue problem.

3.4 Hierarchical SFA Network

Due to the nonlinear expansion of input signals, the input dimensionality increases with the increasing number of inputs. It induces a significant problem for data with high dimensions, for instance, natural images. One ideal solution to overcome this curse of dimensionality issue is to divide the input data into smaller regions and apply Slow Feature Analysis (SFA) to these regions individually. The next step concatenates obtained solutions, which serve as an input for the next iteration of Slow Feature Analysis (SFA). Thus, the application of this approach helps to avoid the dimensionality problem. However, the extracted slow features might not be identical to the global solution obtained with the complete input. This strategy also relies on the locality of feature correlations in the input data. The splitting of the data into smaller regions leads to a hierarchical SFA network, which is similar to the feed-forward organization of the visual system, as shown in the fig. 3.3. The size of the receptive field is bigger for higher layers, so it is possible to extract increasingly complex features at these layers, e.g., encoding information of complete objects.

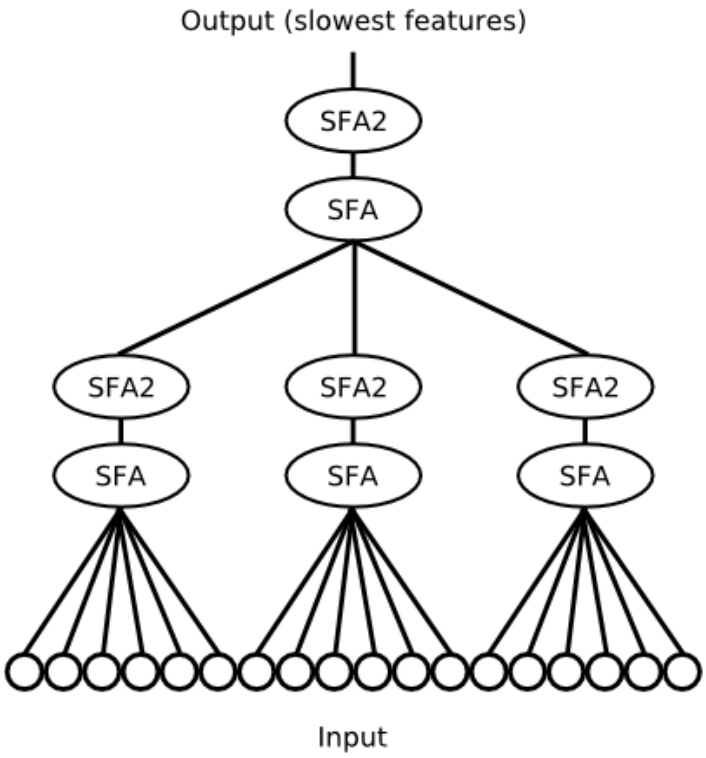


FIGURE 3.3: **Schematic of a two-layer hierarchical SFA network** - Reproduced from Wiskott et al., 2011: Linear SFA is applied to each receptive field for dimensionality reduction, followed by quadratic SFA, i.e., SFA2 (linear SFA after a quadratic expansion).

3.5 Applications in Computational Neuroscience

There are various practical applications of the algorithm, especially in computational neuroscience. It has been applied to the self-organization of complex-cell receptive fields, invariant object recognition, and nonlinear blind source separation (Wiskott et al., 2011). This section describes the relevant application in the context of localization.

3.5.1 Oriospatial Cells in the Hippocampus

The hippocampus is a brain structure important for episodic memory and navigation. Different cell types are present in the hippocampus and neighboring areas, whose responses correlate with the animal's position and head-direction in space. These cells include Grid, Place, Head-direction, and Spatial-view Cells, collectively called Oriospatial Cells. Grid Cells are not spatially localized but show a regular ring activity on a hexagonal grid in space while remaining invariant to orientation. Place Cells are spatially localized and code for the animal's position in space while remaining invariant to orientation. Head-direction Cells are selective for the direction of the animal's head, but they are invariant to its position. Spatial-view Cells only fire

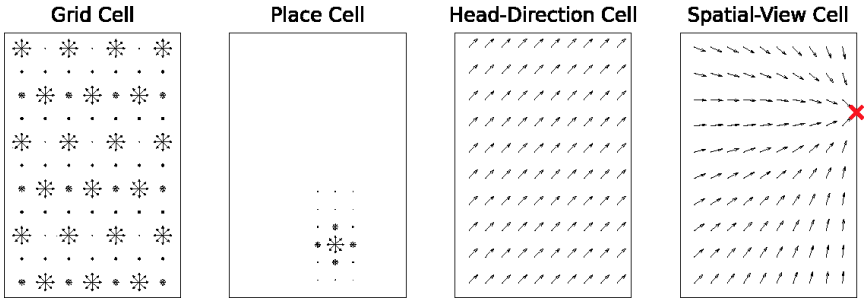


FIGURE 3.4: **Grid Cell, Place Cell, Head-Direction Cell, and Spatial View Cell** - Reproduced from (Franzius et al., 2007): The arrows indicate Oriospatial activity, and their length shows the strength of activity at the arrow base if the animal looks in the direction of the arrow. The Grid Cell is not spatially localized but repeats in a hexagonal grid, whereas the Place Cell is spatially localized, while both are orientation invariant. The Head-Direction Cell's activity shows a global direction preference but is spatially invariant. Spatial-view Cell fire only when a specific view is fixated (indicated by x).

when a particular part of the environment is in the animal's field of view while they are neither orientation invariant nor position invariant. Fig. 3.4 shows the difference between these cells. These Oriospatial Cells probably form the neural basis of an animal's ability to self-localize and navigate (Knierim et al., 1995).

Despite a rapid change of primary sensory signals (i.e., visual input) during an animal's movement in an environment, the firing activity of the Oriospatial Cells changes relatively slowly. This observation forms the basis of a model for the unsupervised formation of such cells based on visual input with the hierarchical SFA network and sparse coding (Franzius et al., 2007). Fig. 3.5 shows the model architecture. It consists of a hierarchical network to overcome the problem of high input dimensionality. The first three layers perform the same sequence of steps, including linear SFA for dimensionality reduction and quadratic expansion of the reduced signals, followed by linear SFA for feature extraction. Linear SFA can effectively find better solutions by a nonlinear expansion of its inputs, similar to the application of nonlinearities in neural networks. However, this expansion increases the feature dimensionality. Therefore, the first step performs dimensionality reduction with linear SFA for computational efficiency. Subsequently, the next step projects the features into the space of quadratic monomials and applies SFA again. The output of the last layer produces a distributed representation of the Oriospatial Cells. The last layer performs an additional sparse coding step to get a localized representation of these cells. The network is trained with visual input as perceived by a virtual rat running through a textured environment. It is easy to consider that the visual input fluctuates faster while the rat changes its position or orientation on a slower time scale. Since SFA extracts slowly varying features, the final representation encodes either position

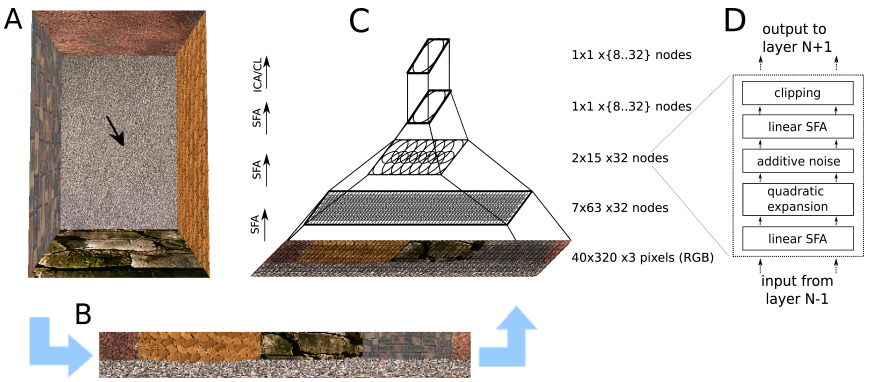


FIGURE 3.5: **Model Architecture** - Reproduced from (Franzius et al., 2007): At a given position and orientation of the virtual rat (arrow) in the naturally textured virtual-reality environment (A), input views are generated (B) and processed in a hierarchical network (C). Lower layers perform the same sequence of steps, including linear SFA for dimensionality reduction, quadratic expansion, additive noise, linear SFA for feature extraction and clipping; the last layer performs sparse coding (either ICA or CL) (D).

or orientation depending on the movement statistics of the rat. So, a purely sensory-driven model can reproduce the firing characteristics of Place, Head-direction, or Spatial-view Cells representing the spatial information in the brain of rats.

3.6 Learning Scene Representation with SFA

This section gives an intuition of how this work exploits SFA to map a robot's surroundings for performing tasks like localization and navigation. For learning a scene representation with SFA, the goal is to find functions representing a robot's position on the $x-$ and $y-$axis as slowly varying features while being independent of other variables like its orientation or condition changes. SFA-based learning depends on temporal statistics of the input data; therefore, if the relevant information (here: robot position) during training changes on a slower time scale compared to other variables, it will be encoded as the slowest features in the learned representations. Consider an image sequence recorded by a robot in an environment, as shown in fig. 3.6. For illustration, if we plot the value of a single pixel over the entire image sequence (red signal), it is apparent that its value changes much faster than the robot's $x-$ and $y-$position over time. This difference in the time scale between the primary sensory input (i.e., pixel value in a video) and the latent information (i.e., the robot's position) will cue SFA to learn a representation that implicitly encodes the robot's position (x,y) as slowly varying features. The work in (Franzius et al., 2007) provides a detailed mathematical analysis of SFA for localization based on the inputs' temporal statistics.

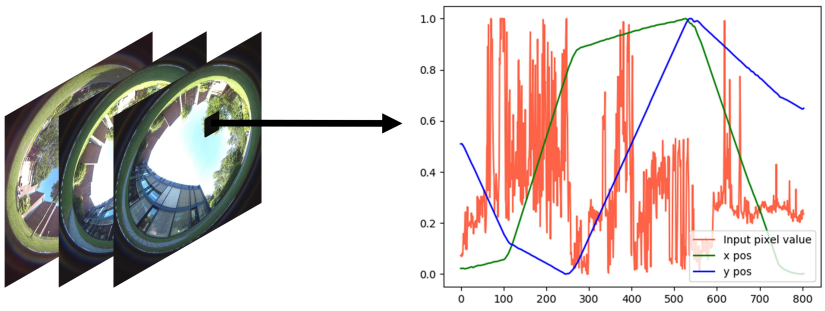


FIGURE 3.6: **Learning Scene Representation with SFA - Intuition**: The figure on the left shows an image stream. On the right, it shows the value of a single pixel plotted for the complete sequence (red) and the robot's $x-$ and $y-$position. The plot shows that the visual input (i.e., pixel value in a video) varies on a faster time scale than the robot's position. This difference in time scale will guide SFA to encode the robot's location (x, y) in the final learned representation as slowly varying features.

Chapter 4

Data Acquisition

A significant contribution of this work is the collection of camera recordings from indoor and outdoor scenes for robot localization. This chapter thus presents the data collection setup and describes the recorded datasets. The experiments were initially conducted in controlled settings using the data generated from a virtual reality simulator for the proof of concept. Later, the same experiments were performed using real-world images to validate the proposed methods' practicality.

4.1 Simulated Data Generation

The artificial datasets used in this work consist of images generated from two simulation software (i.e., *Blender*[1] and *WeBots*[2]). The simulated environment mimics a natural garden-like scene and consists of houses, trees, and man-made objects. A virtual robot traverses an area of size 15×15 meters and captures simulated panoramic views of size 1200×200 pixels. During traversal, it also stores the ground-truth robot position (x, y) corresponding to each image. The data set consists of 15 recordings generated by a random variation of lighting parameters and placing dynamic objects in a scene. The lighting parameters include energy $\in [3, 8]$, the y-coordinate of the light source $\in [-10, 10]$m, and the intensity of the red channel $\in [0.5, 1]$. The dynamic objects include everyday things like cars, bicycles, chairs, and umbrellas. Figure 4.1a and 4.1b shows some images generated from the two simulator packages, respectively. Figure 4.2 shows a robot's traversed trajectory in a virtual scene.

[1] https://www.blender.org/
[2] https://cyberbotics.com/

(a)

(b)

FIGURE 4.1: **Simulated Visual Data:** Images generated from *Blender* (a) and *WeBots* (b).

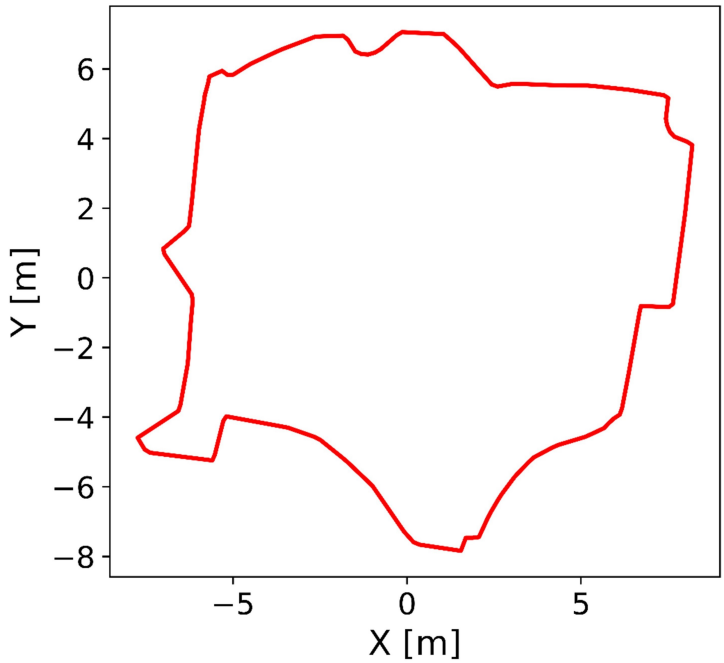


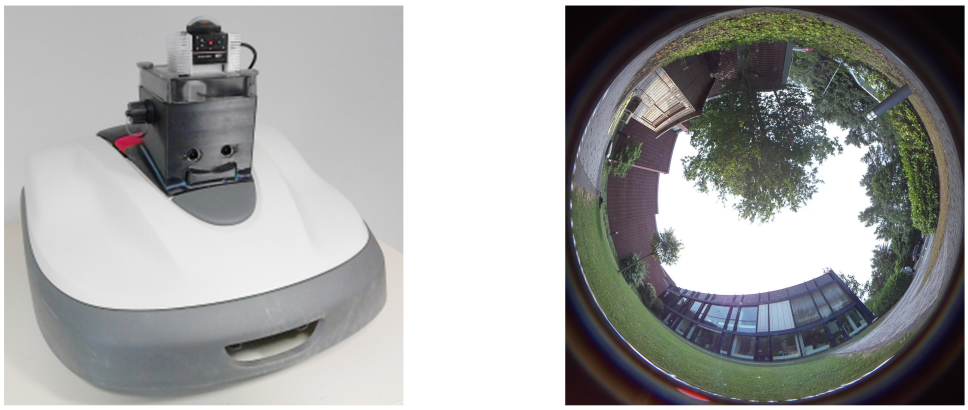
FIGURE 4.2: The robot's traversed trajectory in a simulated environment.

4.2 Real-world Data Collection

This section describes collecting visual data from real-world environments using a modified domestic robot.

4.2.1 Hardware

The real-world recordings were collected from a prototype version of *Honda's Miimo* robotic lawn mower, as shown in fig. 4.3a. It is equipped with a 180° fisheye camera mounted on its top to capture views of an environment. The camera takes omnidirectional images of size 2880×2880 pixels (fig. 4.3b) with a frame rate of 5 fps. The robot also has an integrated ARM processor (i.MX6 ARM board, single-core @800 MHz) and an internal SSD to store the data during recordings.



(a) (b)

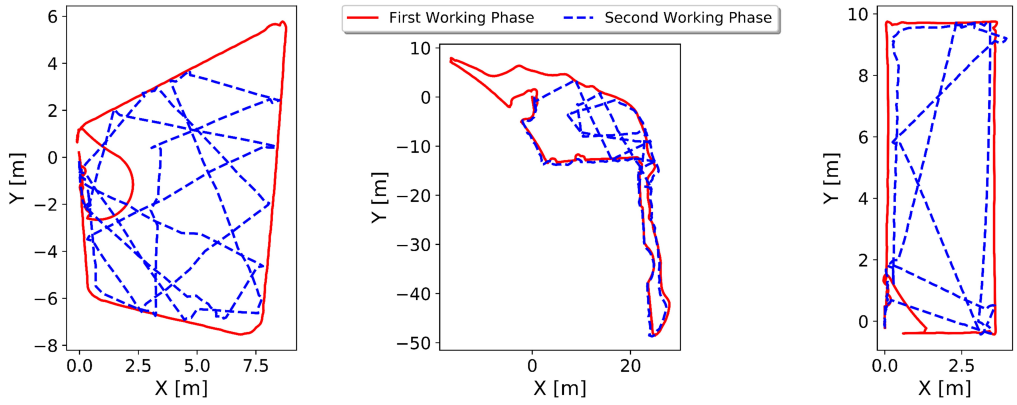
FIGURE 4.3: (a) An autonomous lawn mower robot with a fisheye camera was used for the experiments. (b) shows a captured omnidirectional image.

4.2.2 Recording Environments and Setup

The datasets were recorded from two outdoor scenes and an indoor lab. The unstructured garden-like outdoor environments vary w.r.t their size and complexity. For simplicity, the following text will refer to them as small-scale (9×15 meters) and large-scale (44×57 meters) gardens. In contrast, the indoor lab (4×10 meters) presents a controlled, real-world environment with artificial lighting as the major light source.

Each robot recording session has two operational phases. In the first phase, the robot traverses the border of an area by using the standard wire guidance technology, while in the second phase, it moves freely within the area defined by the border wire. Figure 4.4 shows the robot's trajectories in one of the recording sessions from each

environment. In both working periods, it saves the images and the corresponding odometry information to its onboard storage.



(a) Small Garden (b) Big Garden (c) Indoor Lab

FIGURE 4.4: (a-c) show the robot's traversed trajectories in one of the recording sessions from each environment. The solid line shows the robot's traversal in the first working phase (border run), while the dashed line shows its traversal in the second phase (infield run).

4.2.3 Data Post-processing

The data post-processing includes two steps. The first step unwraps the captured omnidirectional images to their associated panoramic views of size 1200×200 pixels, as shown in fig. 4.5. The second step obtains the ground-truth 2D robot position (x, y) for each image from the odometry information. Please note that the ground-truth data acquisition is necessary to evaluate the performance of the proposed localization methods.

(a)

(b)

FIGURE 4.5: (a) and (b) show an omnidirectional image and its panoramic projection.

Since the robot in the first working phase always starts and ends at the base station by following a wire buried in the ground, it is possible to estimate the robot's ground-truth metric position (x, y) using a method described by Einecke et al., 2018. The authors used wheel odometry and additional weighted loop closure for high-quality localization. However, the technique only estimates the metric shape of the boundary. Thus, the method can not be used to obtain the 2D coordinates (x, y) of the images when the robot moves freely in the second working phase (i.e., infield run). For the second working phase, the ground-truth data (x, y) was obtained using a commercial photogrammetry software, i.e., *Metashape*[3]. It reconstructs a 3D point cloud of the scene and estimates the camera trajectory (i.e., poses from which the images were captured) from a set of unordered 2D image collection (fig. 4.6). The software produces precise localization results (typically in the cm range). However, the procedure to generate camera poses using it takes hours of computation, depending on the dataset. Thus it is not suitable to be used directly for robot localization. After ground-truth data acquisition, the system retained a single image per 2D position (x, y) while discarding the rest of the images for the cases when the robot is stationary.

FIGURE 4.6: **Structure-from-motion (SfM)** - Reproduced from Snavely et al., 2006: It produces 3D scene geometry and camera motion from an unordered 2D image dataset.

4.2.4 Recorded Data Sets

The recorded data sets from the two gardens consist of 25 recordings collected over one and a half years. The recordings contain environmental effects like different daytime, lighting changes, weather, seasons, and dynamic objects. Thus, the image sets cover the most challenging situations a system needs to tackle for successful visual localization. Figures 4.7a and 4.7b show images collected from the small and large-scale gardens. The indoor datasets consist of 10 recordings and present a relatively

[3]https://www.agisoft.com/

constrained environment. These recordings were used for the robot navigation experiments and will be explained later in chapter 9. The figure 4.8 shows some images from the indoor lab.

(a)

(b)

FIGURE 4.7: **Real-World Outdoor Data**: (a) and (b) shows images collected from the small and large-scale gardens.

FIGURE 4.8: **Real-World Indoor Data**: Images collected from the indoor lab.

Chapter 5

Visual Localization and Mapping Pipeline

Visual localization and mapping enable an entity (e.g., robots, self-driving cars) to locate itself in an environment and create a consistent representation of its surroundings using a camera. This chapter presents the proposed pipeline for mapping an environment and using the learned representation to perform localization and navigation.

5.1 Mapping Pipeline

Figure 5.1 shows an overview of the proposed pipeline for learning scene representation using Slow Feature Analysis (SFA). The **input** to the pipeline are omnidirectional images collected by a robot recording session. The **pre-processing** module unwraps the omnidirectional input images to panoramic views and then applies one of the following three pre-processing approaches to the images:
Raw pixel values: The first approach uses the gray-scaled panoramic images without any further pre-processing.
Fourier features: The second approach extracts row-wise Fourier features from the panoramic images and retains the magnitude part corresponding to the lowest 15 frequency components. Representations learned with SFA strongly depend on temporal statistics and much less on modality or pre-processing (Franzius et al., 2007). Extracting Fourier components acts here as a compression that preserves temporal statistics and distinct sensory representations for each position (x,y). A different viewpoint of the same location (x,y) usually degrades localization performance. Extracting Fourier magnitudes from upright images is a simple and effective pre-processing for localization since all views under yaw rotation from the identical position (x,y) yield the same feature vector. Hence, it allows achieving orientation

invariance since the magnitude part of Fourier components is not dependent on robot direction, as shown in (Menegatti et al., 2003; Ishiguro et al., 1996). The compact Fourier representation obtained for each image by the pre-processing approach is used to learn SFA representation.

Landmarks: The third approach identifies landmarks (i.e., salient regions, fig. 5.1) in a scene and then learns spatial representations relative to them. Chapter 7 explains the procedure to recognize and extract the landmarks from the input panoramic images. The system rescales all landmark views to a fixed size before feeding them to the mapping module for learning SFA representation.

The later chapters of this thesis will discuss the pros and cons of each approach. The pre-processing module is the same for training (mapping) and test (localization) images. The **mapping** module uses unsupervised learning capabilities of Slow Feature Analysis (SFA) (Wiskott et al., 2002) to learn the spatial representation of an environment from the training data. Depending on the pre-processing method, the module employs a different SFA network architecture for learning slow features (c.f. 5.1.1). The pipeline's **output** is a trained SFA model, which can then be utilized for the localization and navigation tasks.

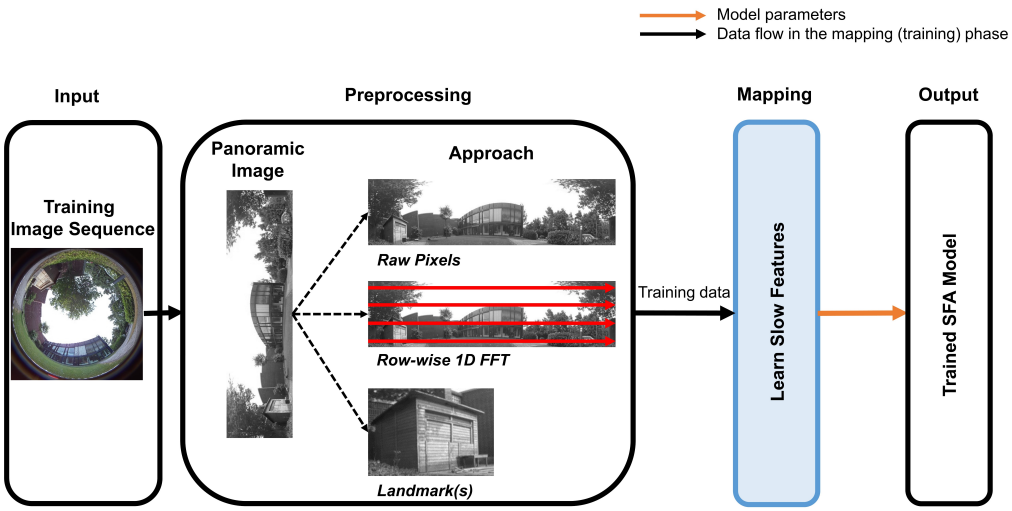


FIGURE 5.1: **Mapping Pipeline**: The input to the pipeline are fisheye views. The pre-processing module projects input images to panoramic views and extracts relevant image representation based on the selected approach (i.e., raw pixels, Fourier Features, or Landmarks). The mapping module uses the training set data to learn the spatial representation using SFA. The output of this pipeline is the spatial representation of an environment.

5.1.1 Network Architecture and Training

As mentioned earlier, the system learns slow features using a different SFA network architecture depending on the image representation. The core SFA learning algorithm is the same for all variants; the only difference lies in the predefined network settings to account for different input dimensionality. Figure 5.2a and 5.2b show the SFA network architecture used for the experiments to learn scene representation using complete (i.e., raw pixels) and landmark images, respectively. The architecture consists of a hierarchical network to cope with high input dimensionality, as learning slow features from images in a single step is computationally infeasible. The hierarchical network consists of four converging layers, where the first three layers have multiple identical nodes arranged on a regular grid and the last layer has a single node. The first layer performs an extra step of patch normalization on input images to mitigate the effects of poor contrast in images due to glare or some other factors. Apart from that, each node in a layer performs linear SFA for dimensionality reduction, quadratic expansion of the reduced signals, and another SFA step to extract slow features. The hierarchical network makes a tiny input region visible to a node, and only this region (i.e., *receptive field*) thus influences a node's output. The size of receptive fields, the distance between the adjacent receptive fields, and expansion dimensionality depend on predefined network settings, which were slightly adapted in the experimentation of this work to account for different image resolutions. However, the hierarchical network, in general, is robust under various parameter settings (Franzius et al., 2007). Tables 5.1 and 5.2 report the network parameter settings for hierarchical SFA on complete and landmark images, respectively. The region of input visible to a node increases with each layer. These layers converge onto a single node whose first eight slowest outputs $s_{1...8}$, also called SFA-output units, were used for the experiments. The first two units $s_{1,2}$ ideally encode the robot's x- and y-position, respectively. The higher units $s_{3...8}$ encode a mixture of the first two units or higher modes. Using a higher number of slow feature output units (i.e., more than the first eight $s_{1...8}$ SFA units) results in overfitting the training data (Metka, 2019). Therefore, this work uses up to the first eight slow features $s_{1...8}$ for localization and navigation tasks.

Network Training: The system trains the layer in an SFA network with all temporally ordered training images. Instead of loading all images once, the system divides the training data into small batches, and each batch is then passed to the layers sequentially. Further, to maintain the continuity between input images, the last image of a batch is re-inserted to the next one. When the training of the nth layer finishes,

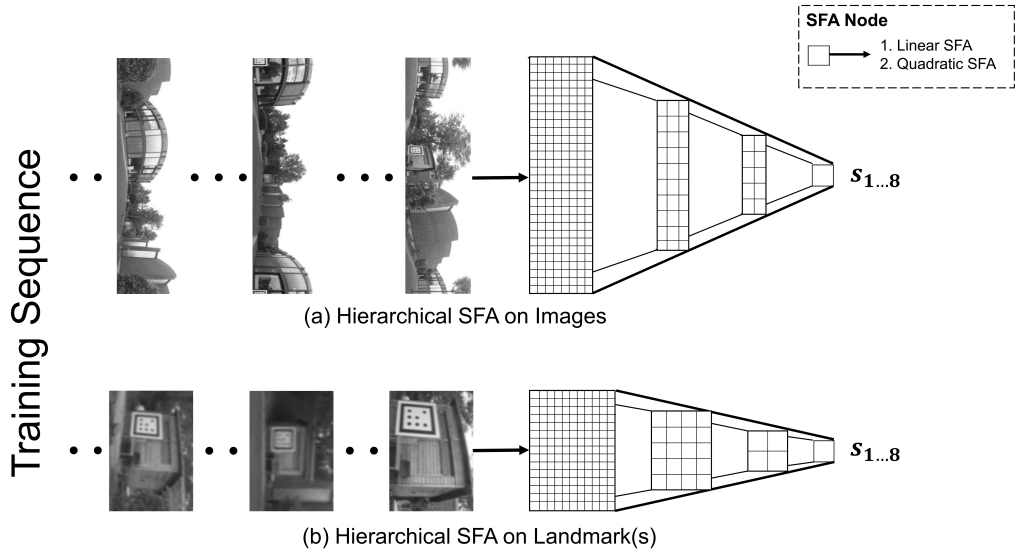


FIGURE 5.2: **Network Architecture for Learning Scene Representation**: Hierarchical SFA on (a) complete images and (b) landmarks. The system processes an image sequence using a four-layer hierarchical SFA network. The two networks differ w.r.t preset parameters to deal with different image resolutions. The layer in a network consists of multiple SFA nodes. Each node performs linear SFA for dimensionality reduction, followed by the application of SFA on quadratically expanded inputs for slow feature extraction. The final node in the last layer outputs an eight-dimensional feature vector, i.e., the slowest features $\mathbf{s_{1...8}}$, which implicitly encodes the learned scene representation.

Layer	RF size (w×h)	Stride (w×h)	Input dim	Output dim
1	12 × 12	6×6	144	12
2	5 × 3	2×3	180	14
3	6 × 3	3×1	252	16
4	1 ×1	1×1	240	8

TABLE 5.1: **Network parameters for hierarchical SFA on complete images**: Size of the receptive field (RF size), stride (receptive field distances), and input-output dimensionality of each node are given for all layers of the SFA network.

Layer	RF size (w×h)	Stride (w×h)	Input dim	Output dim
1	10 × 10	5×5	100	12
2	5 × 5	2×2	300	12
3	4 × 4	2×2	192	12
4	1 ×1	1×1	192	8

TABLE 5.2: **Network parameters for hierarchical SFA on landmark images.**

its SFA outputs are computed for the complete training sequence and fed to the next $(n+1)$ layer as input training data. For computational efficiency, the system trains a single node in each layer with stimuli from all node locations and replicates this node throughout the layer (Franzius et al., 2007). This technique is similar to weight

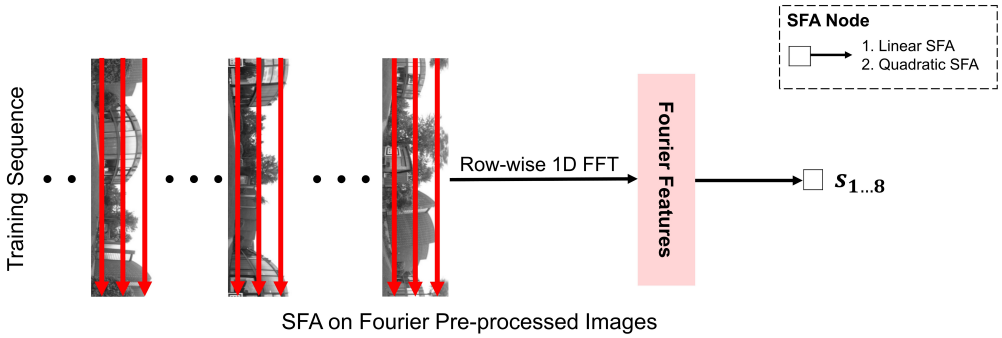


FIGURE 5.3: **Learning Scene Representation using Fourier Pre-processed Images**: The input to the mapping module is the row-wise Fourier features of the training images. In this case, the architecture comprises only a single SFA node and performs two steps to extract slow features. The first step reduces the input dimensionality using linear SFA, which computes the input-output functions as the weighted sum of the input data. The second step extracts slow features by performing 2nd-degree polynomial expansion of the input data. The output is an eight-dimensional feature vector, i.e., the slowest features $\mathbf{s_{1...8}}$, representing the learned scene representation.

sharing in Neural Networks.

Learning Scene Representation using Fourier Features: In the case of Fourier pre-processed images, the system first extracts the row-wise features discussed earlier and then uses these features to learn spatial representation using SFA (fig. 5.3). Thus, in this case, the input to the mapping module is Fourier features of the training images instead of the pixel values. The inclusion of Fourier pre-processing omits the need for a hierarchical SFA network due to low input dimensionality. It allows the processing of the entire input using a single SFA node. Hence, the learning phase only consists of two steps; the first applies linear SFA, and the second extract non-linear slow features using a quadratic SFA. The input and output dimensionality for the first step are 900 and 20, respectively. The input and output dimensionality for the second step are 20 and 8, respectively. The eight SFA-output units $s_{1...8}$ represent the slowest features learned by SFA.

5.2 Localization Pipeline

The mapping pipeline introduced in the previous section learns localization relevant scene representation by applying SFA on input images. The SFA approach generates an implicit map (i.e., mapless representation) of the environment. After mapping an environment, the computation step is instantaneous, i.e., the trained model only requires a single test image to compute the SFA output. The left part (green) of the localization pipeline (fig. 5.4) shows the procedure to obtain a test image's output in

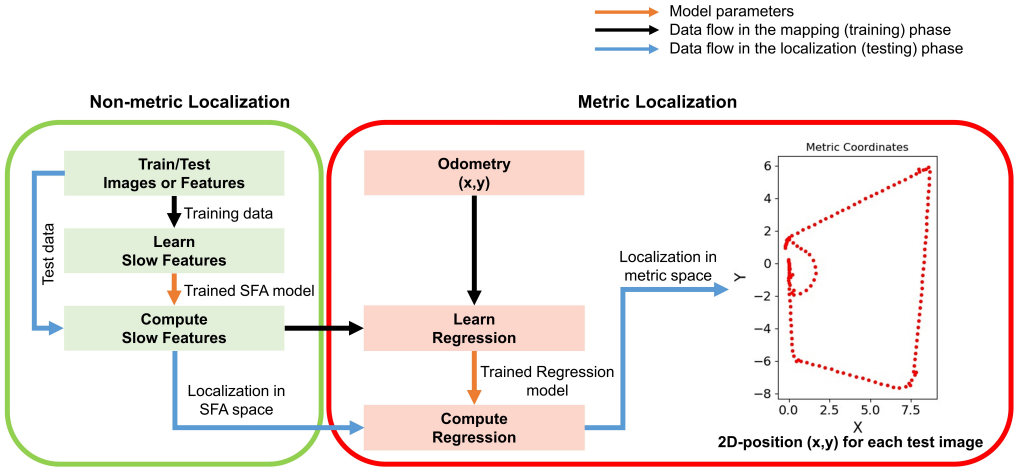


FIGURE 5.4: **Localization Pipeline**: The pipeline shows the procedure to estimate a robot's position (x, y) from an input image. SFA-based localization is instantaneous, i.e., it only needs a single image to compute the model output. Therefore, it takes a test image or features and uses the trained SFA model to localize it in the SFA space (left part). Moreover, this representation space is also sufficient to perform 2D robot navigation. The system uses the odometry data and learned SFA representation to compute a metric regression function for quantifying the model performance in metric space (right part). The last step in the pipeline applies the learned regression function on the SFA output of a test image (i.e., localization in the SFA space) to obtain its 2D (x, y) position.

SFA space. The **non-metric localization** module uses a test image or features and the trained model to perform localization in the slow feature space. In addition, this representation is also sufficient for goal-directed navigation in the SFA space without explicit path planning (c.f. chapter 9).

Although the generated SFA representation is sufficient for localization from single images and 2D robot navigation in the same representation space, we must map this representation into traditional metric (x, y) coordinates to compare it with other localization methods. The right part (red) of the localization pipeline (fig. 5.4) shows how to obtain a test image's output in metric coordinates (x, y) (i.e., **metric localization**). This step is required only to assess the learned spatial representation in metric space. The system projects SFA to the metric space using the learned SFA representation (mapping) and odometry (x, y) data. It trains a regression function that learns to predict the x and y position of a robot from slow feature values. In the testing (localization) phase, the system predicts the 2D position (x, y) for each test image by applying the learned regression function on its SFA output.

Chapter 6

Long-term Robust Localization

This chapter is based on the following peer-reviewed publications:

- Haris, M., Metka, B., Franzius, M., & Bauer-Wersing, U. (2017). Condition invariant visual localization using slow feature analysis. New Challenges in Neural Computation (NC2). Machine Learning Reports, Frank-Michael Schleif.
- Haris, M., Franzius, M., & Bauer-Wersing, U. (2019). Robust outdoor self-localization in changing environments. In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS, pp. 714-719). IEEE.

Vision-based autonomous mobile robots operating for a long time have to deal with changing environments, especially in outdoor scenarios. These changes may come from different lighting conditions, weather, seasonal shifts, or structural change. A drastic change in the environment directly affects a scene's appearance; thus, a single place might look completely different in changing conditions (fig. 6.1). This problem poses a significant challenge for robots that aim to perform long-term visual localization. This chapter introduces the proposed approach for learning invariance to such changes for robust localization and presents the experimental results on simulated and real-world long-term data.

FIGURE 6.1: **Long-term Visual Localization** [1]: The same place can look very different, depending on when it is observed.

[1] https://www.visuallocalization.net/

Several authors have addressed the issue of changing conditions partially by focusing only on illumination changes. The approaches include, for example, an active exposure control method for visual odometry in high dynamic range environments (Zhang et al., 2017), image transformation into lighting-invariant color space (McManus et al., 2014), and visual feature point descriptors (Carlevaris-Bianco et al., 2014). In (Churchill et al., 2013), the system does not build a single monolithic map representing all the observations of a workspace; instead, it creates a composite representation from multiple runs in a workspace to capture the diversity of varying conditions. Despite the system's performance, memory demand and map complexity increase over time. The feature-finding algorithms (Valgren et al., 2010) are well known for performing the task of place recognition but may fail to deal with extreme visual change. Other methods (Sünderhauf et al., 2015) use features from pre-trained deep convolutional neural networks for place recognition. Features extracted from different layers are invariant w.r.t to viewpoint and condition changes. However, the computation and matching of the high dimensional features are computationally expensive, which may not suit real-time operation on a mobile robot. In (Kendall et al., 2017), the authors have used PoseNet (Kendall et al., 2015), which is trained end-to-end to estimate the camera's six DOF pose from a single monocular image. However, it is also computationally expensive and requires ground-truth positional data. State-of-the-art structure-based methods perform well in generalizing to different viewpoints in similar conditions (Sattler et al., 2019) but may ultimately fail in changing conditions (e.g., weather, vegetation effects). On the other hand, learning-based approaches can cope with long-term seasonal changes and outperform handcrafted features (Toft et al., 2022) but require labeled data and high-end hardware (e.g., GPUs) for the training phase.

6.1 Learning Invariance to Short-term Conditions

Unsupervised learning of scene representation with SFA allows visual localization by processing the images captured during a training (mapping) phase. The variable of interest for localization is the robot's position. Suppose during training; it changes on a slower time scale than other variables (for instance, the robot's orientation and environmental changes). In that case, the robot's position will be the slowest feature learned by SFA.

In a controlled setting with no environmental changes, the resulting SFA outputs encode the position or orientation of the robot depending on the movement statistics during training (Franzius et al., 2007). However, it is rare to have fixed environmental conditions in real-world scenarios. If these changes occur on a slower or same

time scale than the robot's position, the resulting representation may encode them as slowly varying features, which can prevent successful localization.

The work in (Metka et al., 2016) coped with short- and mid-term condition changes occurring during the training phase. The proposed method uses a purely data-driven approach and the SFA algorithm to learn the condition invariant representation of an environment. The idea is to restructure the temporal order of the training sequence based on loop closures in a trajectory (fig. 6.2). Images from loop closures represent the same place in different conditions. Thus, whenever the robot revisits a place, the current image is inserted next to the previously recorded image from the same place in the training sequence. In the restructured training sequence, views of the same place in different conditions thus appear temporally close. Hence, it serves as a feedback signal, forcing the SFA model to produce similar outputs at similar locations due to its slowness objective. This technique thus changes the robot's perceived temporal input statistics, allowing learning an invariance to such changes. The experiments (Metka et al., 2016) show learning invariance to the changes occurring during the training phase using the restructuring method. However, using the loop closure events from a single training run limits the usability of the learned representation over a more extended time, i.e., it is valid only if the image statistics between the training and testing phase are similar. The following section describes applying the proposed approach to long-term recordings for robust localization.

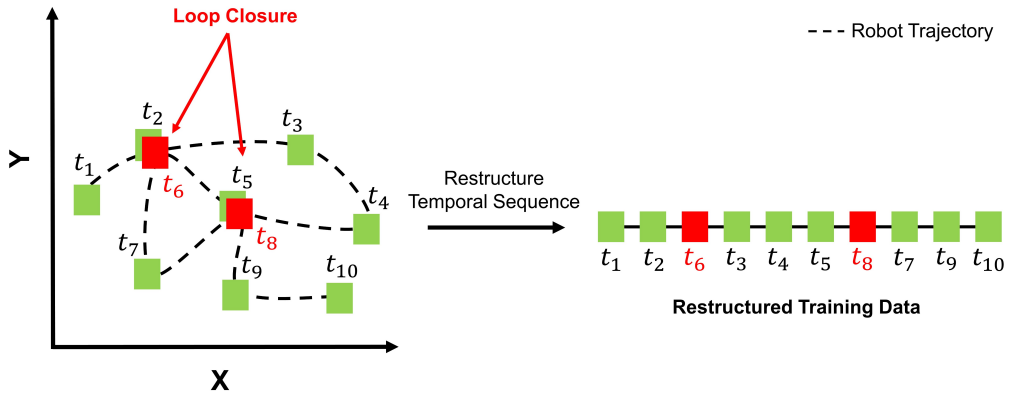
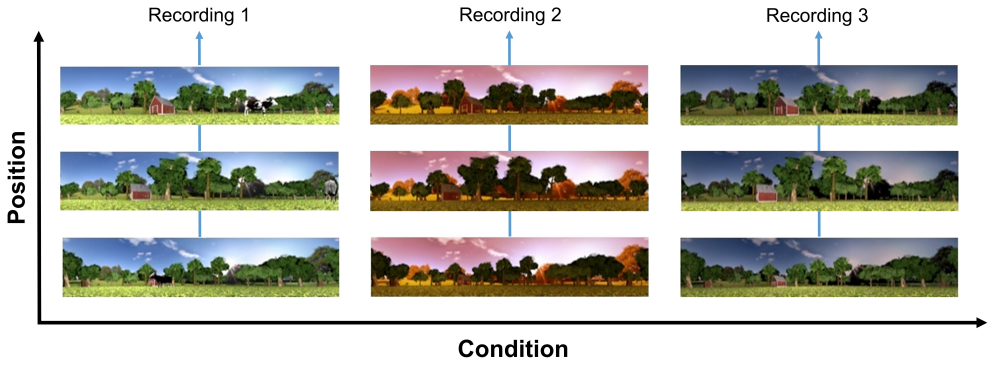


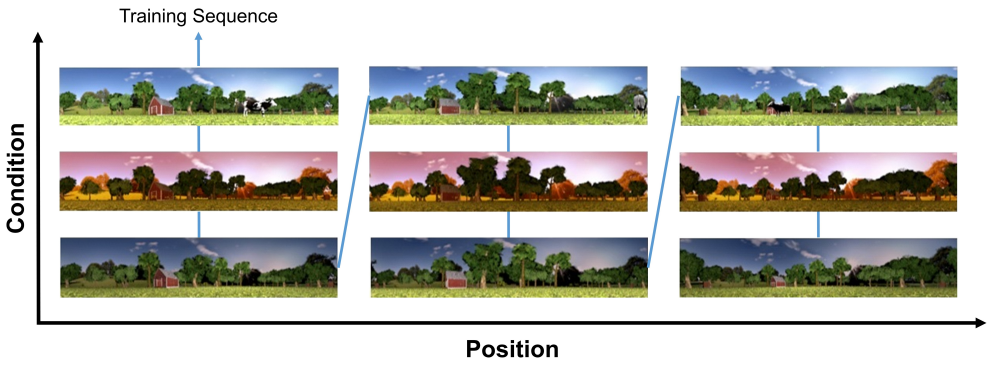
FIGURE 6.2: **Learning Invariance to Short-term Condition Changes**: Figure shows the mechanism to learn invariance to changing conditions during a single training run. Each time stamp t represents an image collected by a robot while traversing a scene, and images from loop closure represent the same place in different conditions. The approach mitigates condition changes by restructuring the training sequence based on loop closure events such that in the resulting sequence, the environment condition changes faster than the robot's position. This restructuring will enable SFA to learn invariance to scene changes and encode only for the robot's position as slowest features in the learned representation.

6.2 Learning Invariance to Long-term Conditions

The goal here is to learn representations that encode the robot's position regardless of the scene changes occurring over a long time (e.g., a year). Thus, this section extends and applies the introduced restructuring scheme to long-term recordings from the same trajectory. In each recording session, a robot automatically traverses a fixed closed-loop trajectory to store images and odometry information. For restructuring the long-term data, the system applies the nearest neighbor search algorithm to the odometry data and establishes position correspondences between the recordings. This association allows inserting images from the same position (x,y) in different conditions into a training sequence before proceeding to the next position (x,y) in a trajectory. In the resulting training sequence, the condition changes will vary faster than the robot's position. Hence, it will enable SFA to learn invariance to long-term environmental conditions while encoding the robot's position. Figure 6.3 illustrates the organization of the training data.



(a)



(b)

FIGURE 6.3: **Learning Invariance to Long-term Condition Changes**: (a) Robot recordings in different conditions. (b) Restructuring Scheme. The training sequence includes images from the same trajectory in different environmental conditions. The joining of long-term recordings based on position (x,y) allows the creation of a training sequence where environmental conditions change faster than the robot's position (x,y), which will enable SFA to learn invariance to such changes.

6.3 Experimental Results

This section presents the experimental results of learning condition invariant visual localization. The proposed method restructures the training data such that undesired variables (e.g., conditions) vary faster than the desired ones (e.g., robot's position) over time. Please note that the method only reshuffles the training data for learning condition invariant representation. The procedure to learn scene representation (mapping) and perform localization is the same as described in chapter 5. The used datasets for the experiments include images from simulated and real-world outdoor environments.

6.3.1 Experiments in Simulated Environments

The experiments were conducted in a simulated environment to verify previously described concepts. The following sections investigate different scenarios w.r.t changing conditions and use specific image sets for each case.

Localization Performance on a Single Recording

This experiment tested the localization performance in a static environment using a single recording. The training set has 279 panoramic images, which were used to train the SFA network (i.e., hierarchical SFA on images). After the training phase, the system retained the first eight slowest features ($s_{1...8}$) as the learned representation. Since all the captured images are from the same environment, to train the regression model for metric mapping, the localization module split the data into a random train and test sets with a split size of 0.70 and 0.30, respectively. This split set and the corresponding coordinates (x, y) are used to train the regression model. The learned regression function is then used to predict the coordinates (x', y') of the test set by applying the function to the SFA outputs of the test set. The localization result has shown good performance as the mean Euclidean distance between the estimated (x', y') and ground-truth (x, y) coordinates is 0.08 meters. Figure 6.4 shows the localization result.

Effect of Slowly Changing Light on Localization

In outdoor scenarios, variables like global illumination might vary on an equal or slower time scale than the robot's position. Such conditions may influence the learned spatial representation and hence reduce localization accuracy. Thus, this experiment aimed to test the impact of lighting changes on localization. For generating the training set, the intensity of the light source was increased such that it changes on a slower

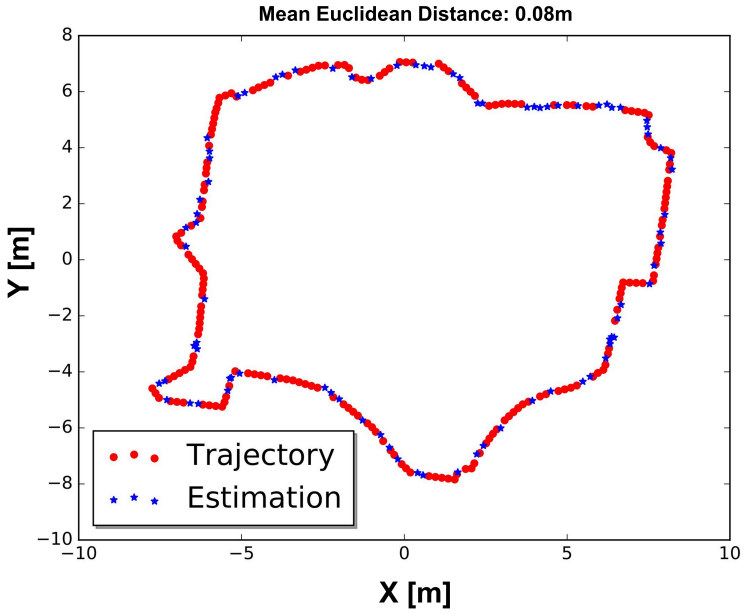


FIGURE 6.4: **Localization in Static Conditions**: The red dots show the training trajectory, while the blue stars show the predicted test locations using SFA localization on complete images. In a fully-controlled setting, it achieved nearly perfect localization, as expected.

time scale than the robot's position during the training run (fig. 6.5). The test set is the same as the previous experiment consisting of no lighting changes. After the learning phase, the system used the training set and its corresponding coordinates (x,y) to train the regression function. The learned function was applied to the slow feature values of the test set images to predict the corresponding spatial coordinates (x',y'). Figure 6.6 shows the localization result. The estimation reflects weaker encoding of spatial information and a localization failure in the west area of the trajectory. The mean Euclidean distance between the ground-truth (x,y) and estimated coordinates (x',y') is 1.41 meters. Thus, the slowly changing light has indeed affected the learned spatial SFA representation. However, the learned representation still encodes some positional information since the estimated coordinates (x',y') are not entirely random.

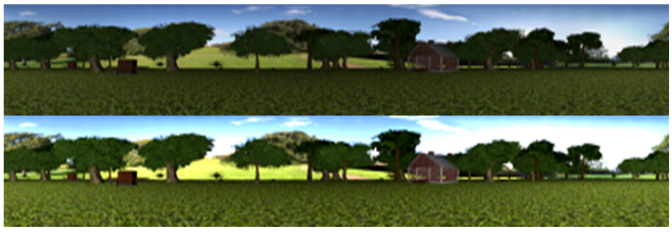


FIGURE 6.5: **Slowly Changing Lighting Condition**: First (Top) and last (Bottom) image of the training sequence; the intensity of the light source was increased throughout the training run such that it changes slowly than the robot's position over time.

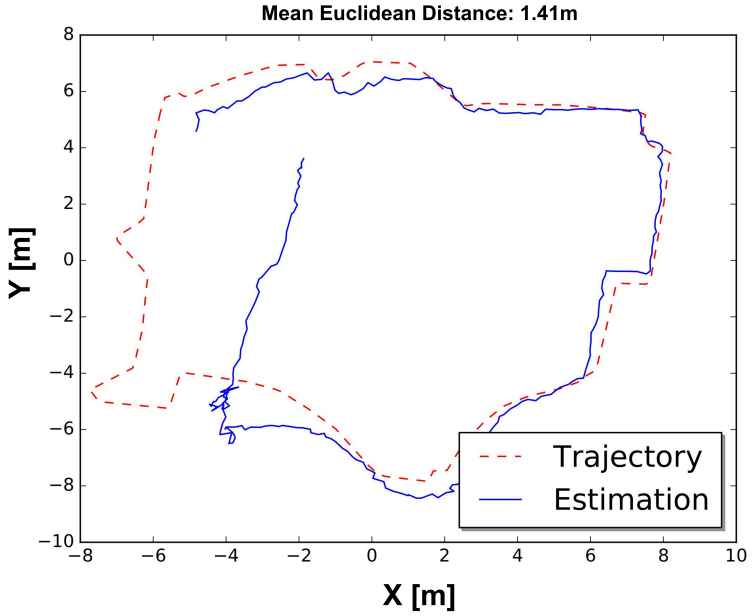


FIGURE 6.6: **Effect of Varying Lighting Conditions on Localization**: Ground truth (red) and estimated coordinates computed by SFA localization (blue). The change in light intensity during the training run has deteriorated the learned spatial representation and resulted in reduced localization performance.

The example presented above is one of the environmental changes which can occur during a training run. However, these changes can vary drastically over a long time, ultimately affecting the localization performance. The following experiment shows the application of the proposed restructuring scheme to deal with the problem. The approach combines data sets from multiple training runs collected over different times based on their position correspondences. This causes environmental conditions to vary faster than the robot's position in the training data, enabling SFA to learn invariance against them. For this experiment, the system joined both sets, i.e., the static light set and the varying light set, to test the impact of the proposed strategy. The learned representation significantly improved the localization accuracy as the mean Euclidean distance between the ground-truth (x,y) and estimated coordinates (x',y') is 0.17 meters (fig. 6.7). It is essential to consider that for long-term robustness; we need more data sets with high variation w.r.t environmental conditions so that SFA always codes for the robot's position in the learned representation.

Influence of Dynamic Objects on Localization

This experiment tested another vital aspect of environmental condition changes. Since natural scenes are always dynamic, this experiment tests the influence of such objects on localization that were not present in the environment during training (mapping

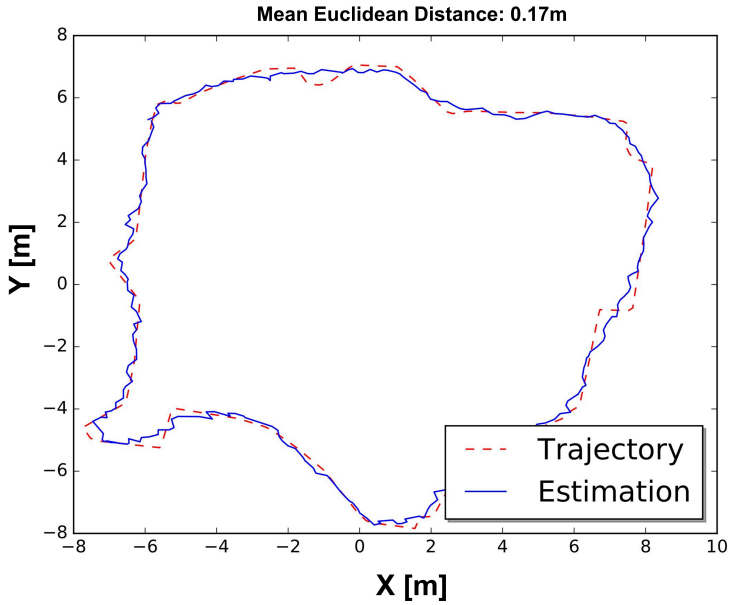


FIGURE 6.7: **Localization with the Proposed Restructuring Scheme**: Ground truth (red) and estimated coordinates computed by SFA localization (blue). Joining the sets in different environments based on loop closures clearly facilitates the SFA algorithm to learn invariance to changing environmental conditions.

phase). Thus, the generated test set for this experiment contains the model of a cow placed in the scene. Figure 6.8 shows an image from the training and test set. The cow's place was kept fixed for the complete test run. Surprisingly, the static object did not affect the learned representation, as the mean Euclidean distance between the ground-truth (x,y) and estimated set (x',y') is 0.12 meters. Figure 6.9 shows the localization result of this experiment.

FIGURE 6.8: **Environmental Change w.r.t a Dynamic Object**: An image from the training set (Top). A test image from the same scene with a cow model as a dynamic object (Bottom).

As evident from the previous simulated experiments, the change in conditions influences the learned representation and thus affects the localization accuracy. The following experiments show how we can use the long-term data sets and the proposed strategy to learn representations invariant to changes occurring over a long

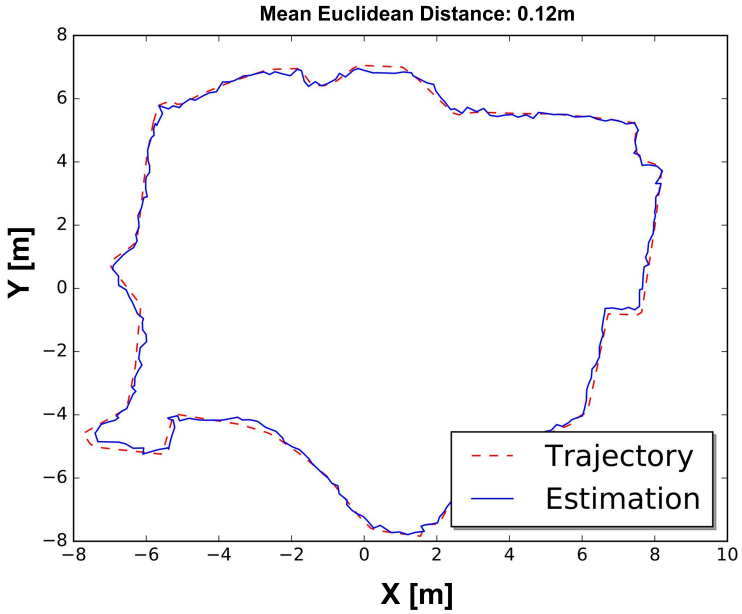


FIGURE 6.9: **Effect of a Dynamic Object on Localization**: Ground truth (red) and estimated coordinates computed by SFA localization (blue). The presence of a single dynamic object does not seem to influence the localization performance.

time. Please note that the results are presented using the complete and Fourier preprocessed images, and the proposed method in this chapter also works with the landmark images.

Learning Invariance to Long-term Environmental Changes

The data sets used for the simulated experiment consist of 10 recordings generated by a random variation of lighting parameters. Each recording consists of 297 panoramic images. Based on position correspondences, the system reordered the training sequence such that the environmental condition varied faster than the robot's position, as already discussed earlier. The SFA model is trained with an increasing number of up to nine data sets and the performance is then tested on the successive set by computing a regression function from the SFA outputs to ground-truth positions (x, y). The same experimental procedure was repeated for 15 random permutations of the image sets to validate the results. Figure 6.10 shows the mean localization performance of the experiments, and figure 6.11 shows an estimated trajectory for the 10th set using nine data sets for training. The mean Euclidean distance for the last test set is 0.5 and 0.4 meters for hierarchical SFA on images and SFA on Fourier preprocessed images, respectively. Using the representations learned in a single condition has a prohibitively significant error in a different condition, but it decreases quickly for more data sets. Thus, the results have shown that it is possible to learn an increasingly invariant representation of the environment.

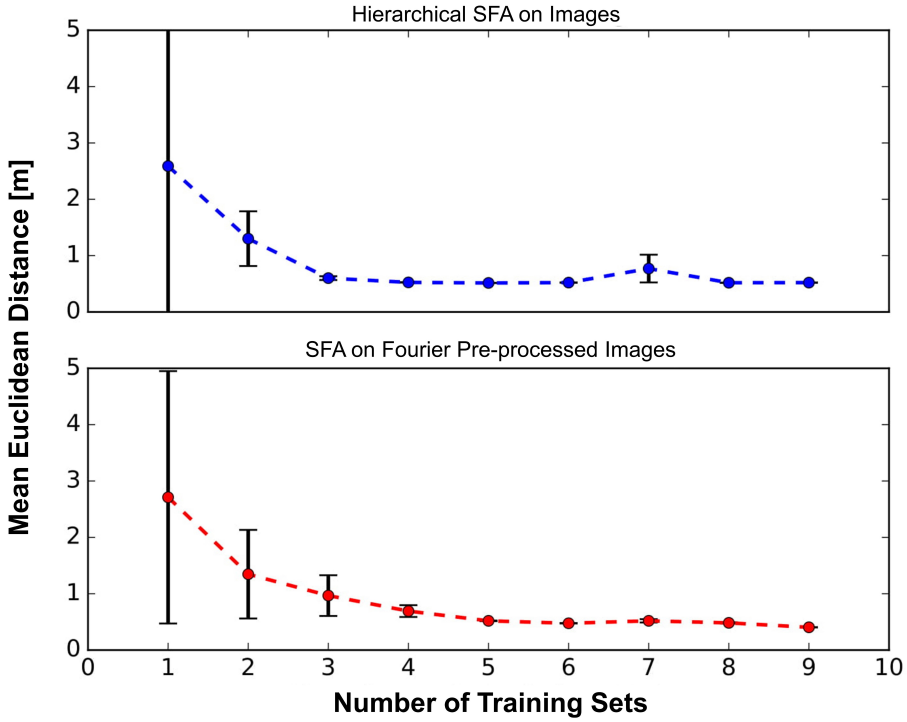


FIGURE 6.10: **Long-term Visual Localization in a Simulated Environment**: Mean and variance localization performance for 15 random permutations of 10 simulated recordings for learning with raw images (Top) and Fourier pre-processed images (Bottom). Each point indicates the localization error for the next unseen condition. Both plots show similar characteristics, an increasingly invariant representation of the environment is possible to learn by adding training sets, which have a high variation w.r.t environmental conditions.

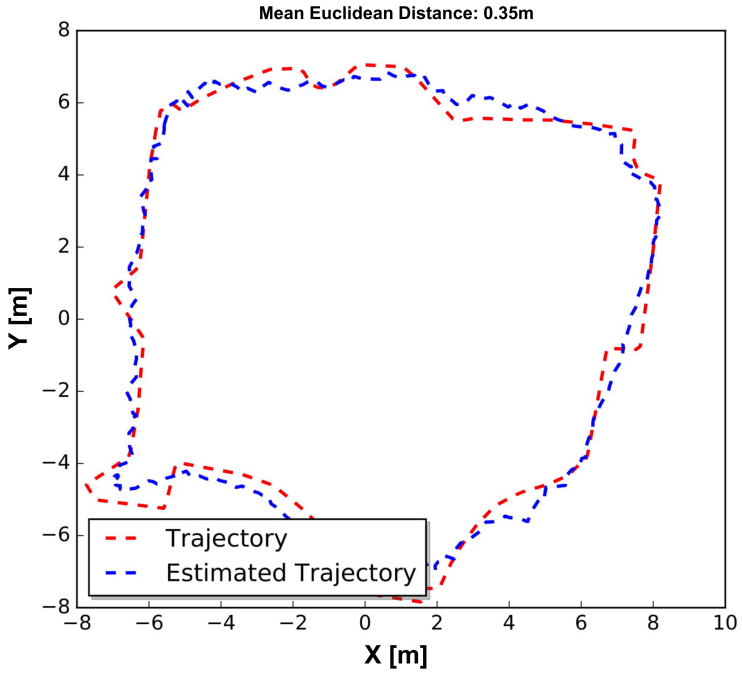


FIGURE 6.11: The estimated trajectory for an unseen set (blue) using nine image sets with a mean Euclidean error of 0.35 meters using the Fourier-based model.

6.3.2 Experiments in a Real-world Environment

This section demonstrates the application of the proposed strategy to real-world outdoor data. The data set has 15 long-term recordings collected over an entire year, i.e., 12 months. These recordings have environmental effects like different daytime, weather, seasons, and dynamic obstacles. The proposed strategy requires combining the long-term sets based on their nearest neighbor position correspondences. After establishing the association, each data set consists of 799 panoramic images. Here, the experiment procedure is the same as in the simulator environment. Figure 6.12 shows the mean localization performance of the experiments, and figure 6.13 shows an estimated trajectory for the 15th set using 14 data sets for training. The mean Euclidean error for the last test set is 0.7 and 1.1 meters for hierarchical SFA on images and SFA on Fourier preprocessed images, respectively. As to be expected, the use of a single training set is not sufficient to perform localization in a different environment as the mean error is relatively high. However, adding successive data sets to the training sequence quickly and noticeably reduces the error. Figure 6.14 visualizes the improvement in estimated trajectory as the model becomes invariant to long-term changes over time.

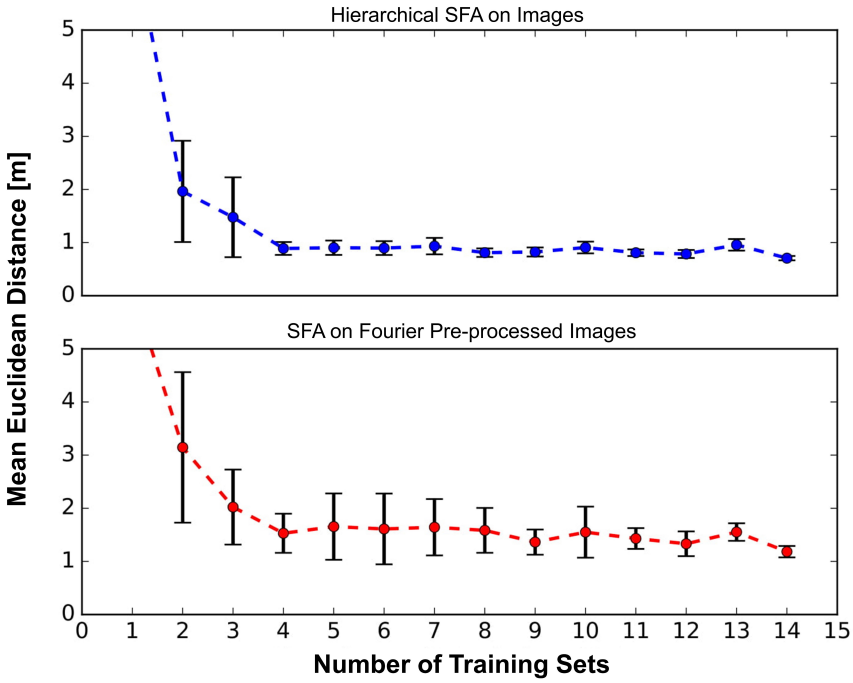


FIGURE 6.12: **Long-term Visual Localization in a Real-world Environment**: Mean and variance localization performance for 15 random permutations of 15 real-world recordings for learning with raw images (Top) and Fourier-preprocessed images (Bottom). Each point indicates the localization error for the next unseen condition. The resulting plots show a decrease in localization error by adding the number of training sets with different conditions.

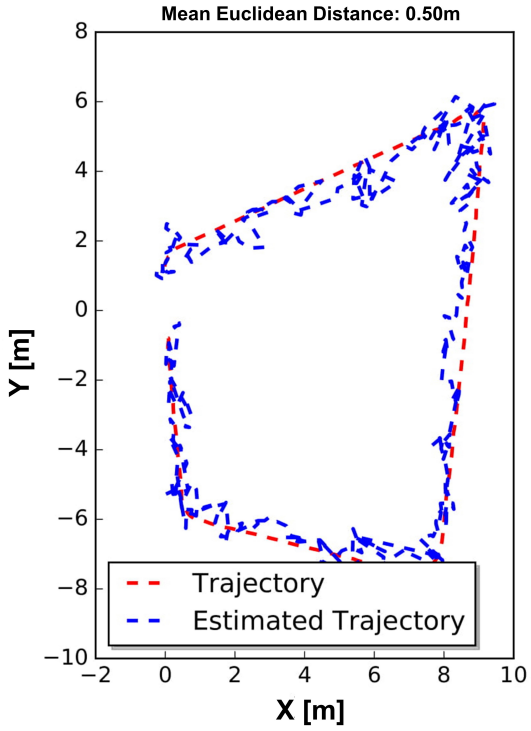


FIGURE 6.13: Estimated trajectory for an unseen set (blue) using 14 image sets with a mean Euclidean error of 0.50 meter using the hierarchical SFA model.

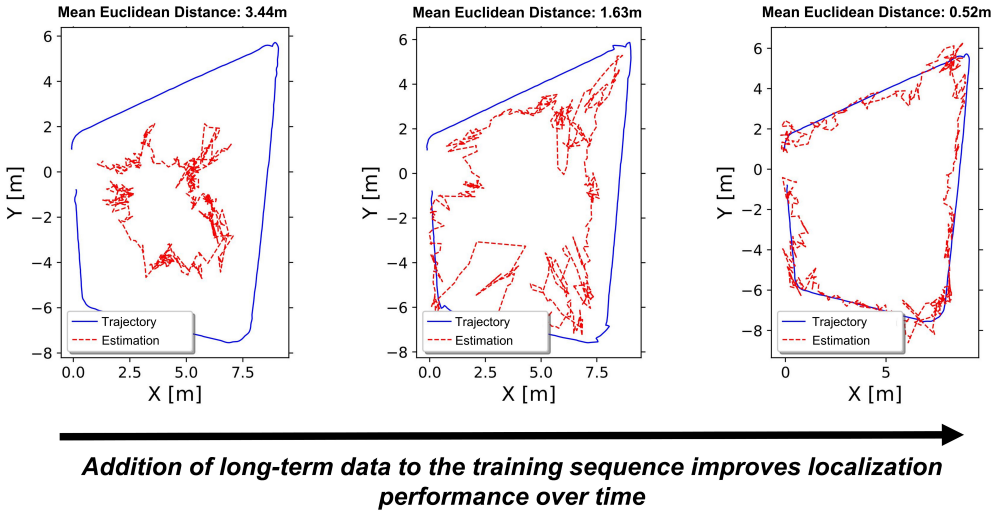


FIGURE 6.14: **Long-term Condition Invariant Learning Improves Localization**: Figure shows the effect of learning environment invariant spatial representation on localization. As expected, it is hard to generalize from one condition to a significantly different condition (left). However, adding more diverse training data enables SFA to learn condition invariant representation and, as a result, allows a robot to robustly localize itself over longer time (right).

6.4 Conclusion

This chapter presented an approach to address the problem of localization in changing conditions. The proposed method reordered long-term data sets based on their position correspondences such that environment conditions vary faster than the position of the robot in the training sequence, allowing SFA to learn invariance to these changes. Learning an invariance to such changes enables a robot to localize itself robustly over a long period from single images. The learned representation was obtained using raw and Fourier pre-processed images. However, the presented restructuring scheme applies to landmark images as well. The Fourier pre-processing step drastically reduces the computation time (c.f. table 6.1). A complete training session with raw images takes 100 s on a standard **CPU (Intel i5 – 6500 with 3.20GHz)** , and localization from a single image takes 165 ms. With the Fourier-based approach, the respective times are 2.85 s and 1.1 ms, which is a significant improvement. Similarly, on an **ARM processor (i.MX6 ARM board, single-core @800 MHz)**, the localization from a raw image takes 2.85 s, and Fourier preprocessed image takes 20 ms. However, this improvement comes with a slight loss in localization accuracy as the mean error for image-based learning is 0.7 meters and for Fourier-based learning is 1.1 meters (c.f. long-term localization experiment, fig. 6.12).

Input to SFA	**Training Time Mapping**	**Execution Time Localization**
Raw Images	100s	165ms
Fourier pre-processed Images	**2.85s**	**1.10ms**

TABLE 6.1: **Computation Times for Mapping and Localization on a standard CPU**: Using Fourier pre-processed images significantly reduces the computation time for learning scene representation and performing localization. Thus, it is well-suited for robots equipped with low-cost embedded hardware.

The proposed method can cope with long-term instability from various sources (e.g., lighting, weather, and seasons) with a straightforward model independent of any feature descriptors. Further, using Fourier representations allows one to do it very efficiently. The experiments have shown that an agent can autonomously learn an invariant representation of the environment. Although the appearance of an environment increasingly changes with seasons, localization accuracy improves over time. Moreover, the efficient computation allows the implementation of the method even on a resource-constrained robot.

Chapter 7

SFA Localization on Landmark Views

This chapter is based on the following peer-reviewed publications:

- Haris, M., Franzius, M., Karanam, K. S. K., & Bauer-Wersing, U (2020). Visual Localization and Mapping with Hybrid SFA. Conference on Robot Learning (CORL), 2020. Proceedings of Machine Learning Research.
- Haris, M., Franzius, M., & Bauer-Wersing, U. (2022). Learning Visual Landmarks for Localization with Minimal Supervision. 21st International Conference on Image Analysis and Processing (ICIAP, pp. 773-786).
- Haris, M., Franzius, M., & Bauer-Wersing, U. (2022). Physical Interactive Localization Learning. In IEEE International Conference on Advanced Robotics and its Social Impacts (ARSO). IEEE

Instead of using complete images, an alternative is to perform visual localization relative to landmarks present in an environment. Landmarks are salient regions that are easier to distinguish in a scene. This approach allows scaling the localization method since each landmark gives an independent estimate of the robot's location. Thus combining several landmarks for localization can offer improved robustness and accuracy compared to using complete images. Moreover, incorporating landmarks into the mapping pipeline enables the generation of semantically meaningful maps, allowing the robots to interact better with the world around them. Artificial landmarks (fig. 7.1a), e.g., ArUCo markers (Garrido-Jurado et al., 2014), are the fastest and easiest to recognize in images. However, it requires a human operator to place such landmarks in a scene before the actual operation, which may not be suitable for many real-world applications except in industrial settings. On the other hand, natural landmarks (e.g., building corners, tree trunks) exist habitually in an environment (fig. 7.1b), thus offering an attractive alternative to fiducial markers. The landmark-based localization approach presented in this chapter employs natural landmarks present in a scene.

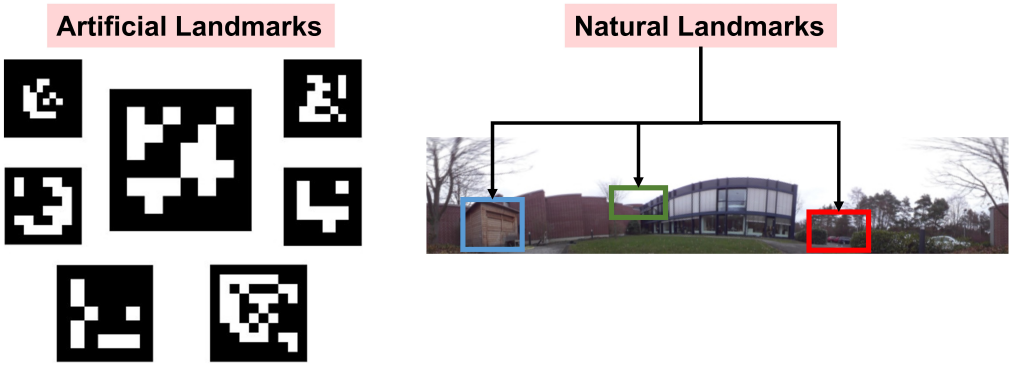


FIGURE 7.1: **Landmarks**: (a) Artificial Landmarks[1]: These landmarks are synthetic markers made up of binary patterns that uniquely encode their id. The markers are straightforward to detect and distinguish in a scene. (b) Natural Landmarks: These landmarks consist of objects or regions naturally present in a scene.

7.1 System Overview

This section presents a high-level overview of learning a scene representation using landmark images. The difference to other image representations presented earlier (i.e., raw pixels, Fourier features) is that we must first detect the landmarks present in a scene before using them for the mapping and localization tasks. Thus, the system uses deep-learning-based CNN (Convolutional Neural Network) detectors to recognize landmarks in the images. These detectors outperform the performance of traditional computer vision algorithms for object detection. However, the following two issues hinder directly using a pre-trained state-of-the-art detector like YOLOv3 (Redmon et al., 2018) for this task:

1. State-of-the-art CNN-based object detection algorithms (e.g., YOLO) are typically trained on MS-COCO objects (Lin et al., 2014). These objects are typically dynamic (e.g., cars, bicycles, persons) in an environment (fig. 7.2a). Localization relative to such objects would break if they are no longer present or moved to another location in a scene. Thus, these objects can not be used as stable landmarks for localization.

2. Even for the pre-trained stable object categories, such objects are not guaranteed to be present in the scene beforehand (fig. 7.2b).

Both these problems make it necessary to learn scene-specific landmarks for most landmark-based localization approaches. Providing sufficient landmark detectors

[1] https://docs.opencv.org/4.x/markers.jpg
[2] https://zhanghanduo.github.io/post/yolo1/

(a) (b)

FIGURE 7.2: **Pre-trained Objects as Landmarks**: (a) Some pre-trained object categories from the MS-COCO data set detected by YOLO-v3[2]. These objects are dynamic and thus can not be used as landmarks for localization. (b) Images from scenes that do not contain any pre-trained objects.

a priori is challenging as suitable landmarks should be long-term detectable and unique. Thus, the landmarks need to be learned on-site. Learning custom landmarks requires labeled training data to fine-tune a pre-trained network (i.e., YOLOv3) or learn a detector from scratch. Hand-labeling is the most prominent method to generate labeled training data, where a human manually annotates objects or regions of interest in hundreds or thousands of images (fig. 7.3). However, this method is time-consuming and costly; hence, it becomes infeasible for large-scale environments with many landmarks.

FIGURE 7.3: **Manual Labeling for Labeled Data Generation**: This technique requires a human to manually specify the objects or regions of interest in hundreds of images for generating labeled training data.

To account for all the problems mentioned so far, the proposed pipeline for performing SFA localization on landmark views consists of the following two stages:

1. The first stage learns visual landmarks in a scene with a new proposed approach that only requires minimal human intervention for generating labeled training data.

2. The second stage uses the learned landmarks and performs localization relative to them.

The following sections of this chapter will explain the individual modules used to perform SFA localization on landmark views. Afterwards, the chapter presents the experimental results of landmarks-based localization using simulated and real-world images.

7.2 Cooperative Landmark Learning

This section presents a novel approach that uses physical object instances with pre-trained visual detectors (e.g., MS-COCO objects) as a labeling tool to learn new stable landmarks for localization. For the sake of simplicity, the following text will refer to objects or regions with pre-trained detectors as ***anchors*** and the derived objects or regions to be learned as ***landmarks***. The idea is to place an anchor in spatial relation to a landmark to teach the robot landmark recognition. Afterwards, the robot collects views from varying perspectives, detects the anchor, infers the landmark position from the anchor position, and generates the annotated image data for learning the new landmark (fig. 7.4). The benefits of the proposed approach are two-fold. Firstly, it enables selecting long-term stable landmarks for localization, and secondly, it allows efficient and fast generation of labeled training data compared to a cumbersome hand-labeling process. Please note that, as mentioned earlier, the pre-trained MS-COCO objects (i.e., anchors) are not suitable for long-term localization since they are typically dynamic. Nevertheless, these objects can serve as mobile teaching anchors for learning new landmarks in a scene.

7.2.1 Method Description

The core idea of the proposed method is to use physical objects in a scene with pre-trained visual detectors as a labeling tool for learning new landmarks. It can be achieved by using or placing such known objects in spatial relation (e.g., next to or under) landmark instances that shall be learned to be detected. Fig. 7.4 shows the steps to learn new landmarks for localization cooperatively with a human. Consider our goal is to learn the painting on the wall as a new landmark. A human interactor places the anchor (i.e., a bicycle) below it to teach the robot landmark recognition. Please note that the anchor's location in the scene remains fixed during the recording phase. The robot now traverses in the environment to collect views from varying perspectives and applies a pre-trained bicycle detector (i.e., YOLOv3, Redmon et al., 2018) to detect its instance in the recorded images. The next step is to specify the spatial relationship of the new landmark w.r.t anchor. At this step in the learning phase, *minimal human supervision* is necessary, i.e., a human now only needs to

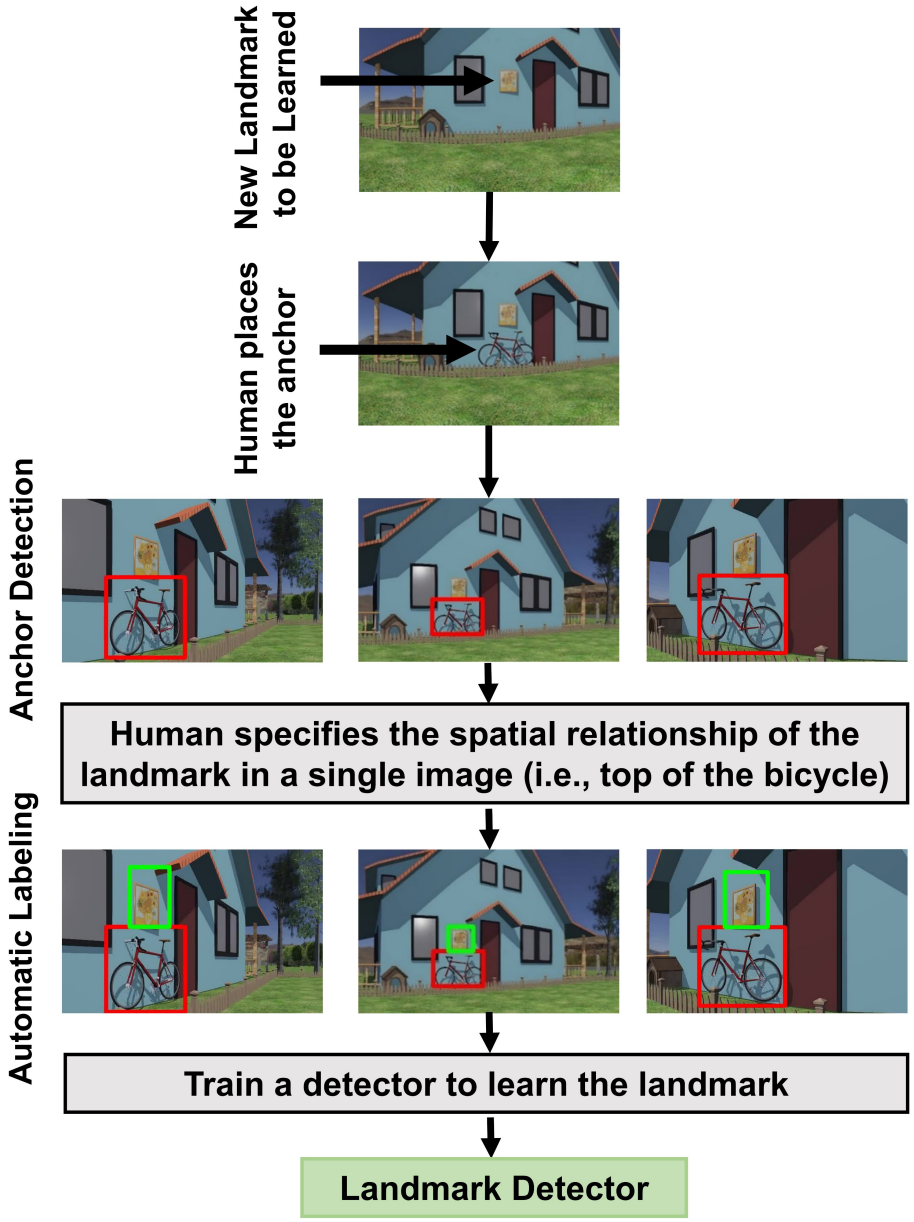


FIGURE 7.4: **Cooperative Landmark Learning**: A human puts a pre-trained anchor (i.e., a bicycle) in a scene to facilitate labeled data generation for learning a new landmark (i.e., painting). The robot then derives and collects imagery from different viewpoints. Afterwards, the system applies the pre-trained detector to recognize the bicycle in the images. A human now specifies the spatial relationship of the new landmark (i.e., the top of the bicycle) in a single image. Next, the system generates the labeled training data from all available images where the bicycle is detected. Finally, the system uses the generated labeled data to train a CNN for landmark detection. The pipeline output is a custom landmark detector that can be used for detecting the instance of this landmark in the images during mapping and localization.

indicate the spatial relation of the bicycle relative to the landmark once to generate labeled training data for this landmark from all recorded images. This relationship is specified as the fixed 2D offset (i.e., above, below) in a single image to derive a

landmark w.r.t an anchor. Thus, the method bootstraps the landmark learning process and removes the need to manually label large amounts of data. The system then uses the instances of the detected anchor and the specified offset to automatically annotate the landmarks in the rest of the recorded images. If YOLOv3 fails to recognize the anchors in some images, the system runs an object tracker to obtain the bounding boxes of anchors in the missing frames. The next step uses the generated labeled data and trains a detector to recognize new landmarks. This step's output is a custom landmark detector, which the robot uses as an independent module for the localization and mapping phases. This process can be repeated to learn more landmarks. The anchor is no longer required after the landmark learning phase and thus can be removed optionally from the scene. The basic implementation of this approach learns one landmark per detected anchor. As an extension, it is possible to scale the system to learn multiple landmarks per anchor, making it straightforward to scale the system without increasing the number of anchors. A human has complete control over the selection of new landmarks. Thus, landmarks can be made unique, robustly detectable, and semantically meaningful, allowing task-relevant localization accuracy in different regions and interactive learning in a playful way.

7.3 Localization Learning on Landmark Views

This section presents the steps to use the learned landmarks to extract the scene representation and perform localization relative to them.

7.3.1 Acquiring Landmark Views

Figure 7.5 shows the steps to detect and extract landmark views. The input is an image sequence obtained by the robot exploration phase in an environment. The system then applies the learned detector from the previous step to recognize the landmark in the collected images. Afterwards, the system resizes each landmark's bounding box to be the same size as the biggest bounding box in its category and rescales the extracted image patch to 120×120 pixels. The output of this step generates an image stream that contains landmark views. The procedure is the same for the images of training (mapping) and test (localization) phases.

7.3.2 Mapping Phase

The mapping module learns the spatial representation of an environment relative to each landmark. It uses the four-layer hierarchical SFA network, which has already been described in chapter 5. The network learns scene encoding in an unsupervised

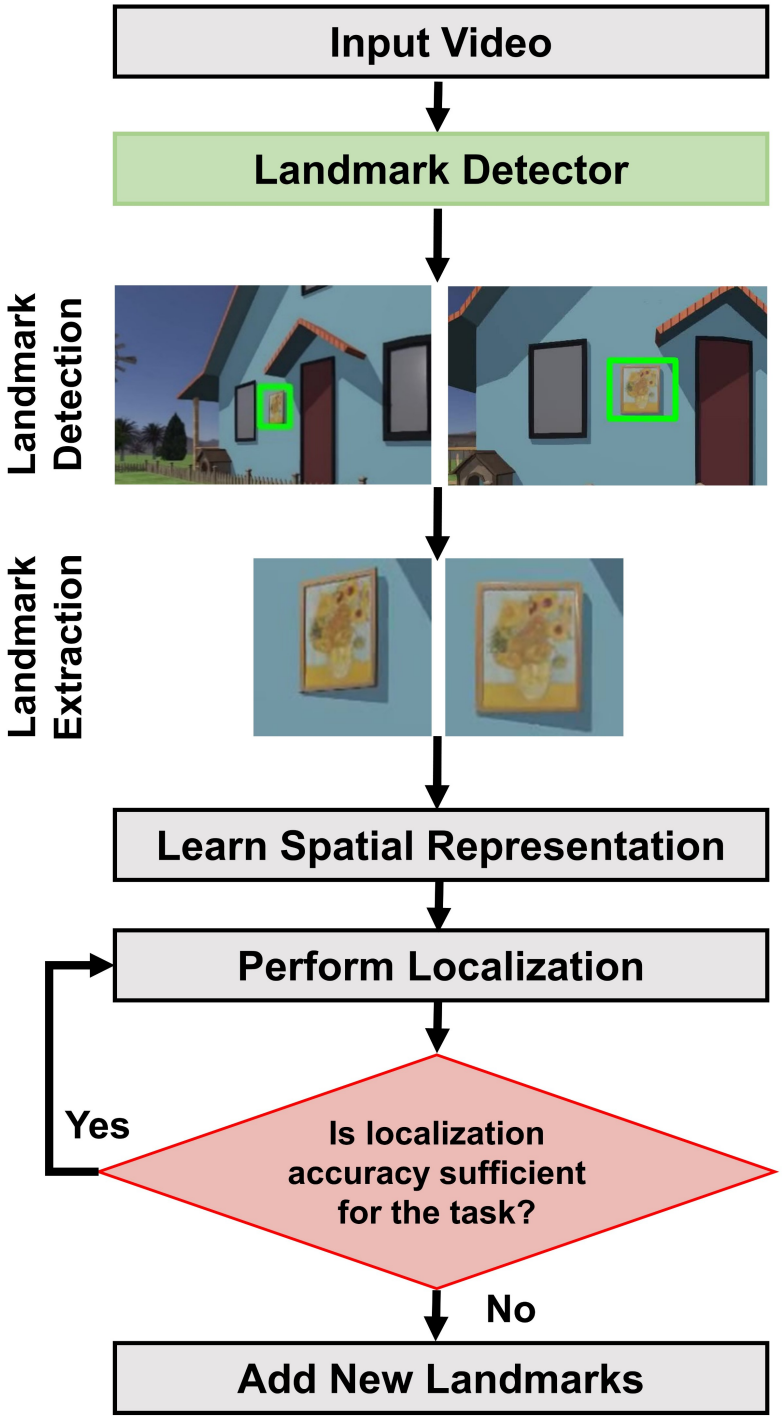


FIGURE 7.5: **Localization Learning on Landmark Views**: The robot uses the learned detector to identify the new landmarks in the images collected for the mapping and localization phases. The next step extracts the landmarks from the images and resizes them to a fixed size of 120 x 120 pixels. The robot then learns spatial representation relative to the landmark and performs localization. If the localization accuracy is not sufficient, the robot can improve the performance by learning multiple landmarks w.r.t to an anchor or can ask a human to place new anchors in the regions where a higher performance is desired.

learning process. This step's output is the compact place representation (i.e., trained SFA model) relative to each landmark.

7.3.3 Localization Phase

The localization module first obtains metric space representation by computing a regression function from the learned spatial representation and odometry data (c.f. 5.2), i.e., the robot's ground truth-position (x, y). This step is performed separately for each landmark; hence, each landmark serves as an independent estimator of the robot's position (x, y).
Afterwards, the localization module uses the learned position estimators to obtain the robot's 2D position (x, y) relative to each landmark. Further, it estimates the robot's global 2D position (x, y) by combining each landmark's position estimation using weighted averaging. The system determines the weight of each landmark by taking the inverse of its localization error. Suppose the robot detects that the localization accuracy is insufficient to execute a task successfully. In that case, it can ask a human to put a known object at some position so that it can learn new landmarks. Hence, the proposed method allows to interactively learn new landmarks in a scene with a human in the loop and scale localization accuracy according to the needs.

7.4 Experiments

This section presents experimental results from simulated and real-world outdoor environments. The system consists of two stages. The first stage learns new landmarks in a scene using the proposed method. The system derives a single landmark for each anchor present in a scene by setting a fixed 2D offset. Afterwards, it uses 500 labeled images for each landmark and trains a detector to learn these landmarks. If the anchor and the landmark are not within the same plane, a simple 2D offset will not capture a semantic region but a subset of the scene's viewing space. However, experimental results suggest that these views can be classified with a CNN, and thus no degradation of localization accuracy on such views is expected. The second stage uses the learned landmarks to perform localization relative to them. This stage proves that the learned landmarks in the first stage are well suited for the localization task. In addition to learned landmarks, the section provides localization accuracy obtained with PoseNet (Kendall et al., 2017) and SFA localization on Fourier pre-processed images for baseline comparison. PoseNet is an end-to-end learning-based visual localization method that directly regresses an image's pose (i.e., position and orientation) from individual images. Like SFA, PoseNet also has an offline learning phase (mapping)

that allows a straightforward comparison. To obtain PoseNet results, the system used 25% of the data (subsampled from the training set, i.e., every 4th image) for validation and the remaining to train the network. Median accuracy is used for comparison to remove the effect of outlier position estimates (x, y) obtained with the baseline method (i.e., PoseNet).

7.4.1 Simulated Experiments

The experiments were performed in a simulated garden with an area of 18×18 meters. A robot randomly traverses the area to record images for the training set and then samples a regular grid to collect the test set. The training and test trajectory consists of 15000 and 1250 panoramic images, respectively. The virtual environment consists of three different CNN-detectable anchors (i.e., a bicycle, a car, and an umbrella). The system then learns one landmark relative to each anchor by training a CNN. After training, the learned landmarks achieved nearly perfect recognition rates (c.f. table 7.1). The system then uses the landmarks to perform localization. Table 7.1 reports the median localization performance w.r.t landmarks and the baseline methods. The combination of landmarks achieves better localization accuracy than the baseline methods in this experiment.

Landmark-based Localization					**Fourier-SFA**	**PoseNet**
Id_1	**Id_2**	**Id_3**	**Combined**	**%**		
0.18m [99 %]	0.27m [99 %]	0.27m [97 %]	**0.16m**	100	0.26m	0.21m

TABLE 7.1: **Localization Results on Simulated Data:** Median localization performance w.r.t learned landmarks and the baseline methods. The table further reports detection rates of the learned landmarks. The combined detection rate of 100% indicates that at least a single landmark is present at any given test location. Localization w.r.t learned landmarks outperforms the baseline methods in this experiment.

7.4.2 Real-world Experiments

The real-world experiments were performed in a small- and large-scale garden-like environment. The small garden work area is $88m^2$, while the big garden work area is $494m^2$. During recordings, an autonomous robot traverses a fixed closed-loop trajectory in both environments (fig. 7.6a) to record the imagery for the experiments. The procedure for collecting the recordings and post-processing has already been described in chapter 4. The collected datasets consist of three recordings from each garden that vary w.r.t daytime, lighting conditions, and dynamic scene changes.
Learning spatial representation relative to landmarks in a scene requires training a detector to recognize the landmarks. Therefore, the system first generates the labeled

training data using the fixed 2D offset between the anchor and the landmark. Figure 7.6b shows an example of the anchor (i.e., a suitcase) and the learned landmark (i.e., an electrical cabinet) present in one of the outdoor environments. Afterwards, it uses the labeled data and fine-tunes YOLOv3 (Redmon et al., 2018) for this task using one of its implementations (Muehlemann, 2019). Please note that the images used for fine-tuning the network belong to separate recordings, i.e., these recordings were not used for mapping and localization phases. After fine-tuning, the system used the detector as an independent module for detecting the landmarks in images. Figure 7.6c shows some learned landmarks from outdoor environments, and figure 7.7 visualizes the detections in one of the recordings from both gardens. After the landmark learning phase, the next step includes learning scene representation (mapping) and performing localization relative to the learned landmarks. The following subsections present the experimental results obtained with small- and large-scale garden datasets.

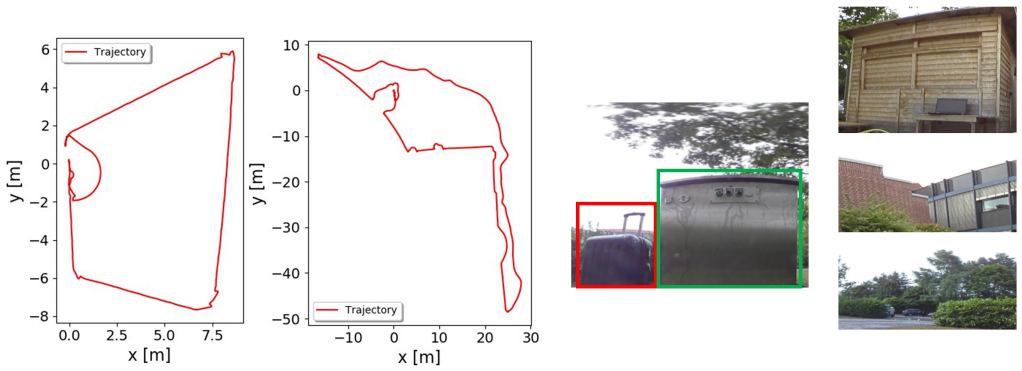
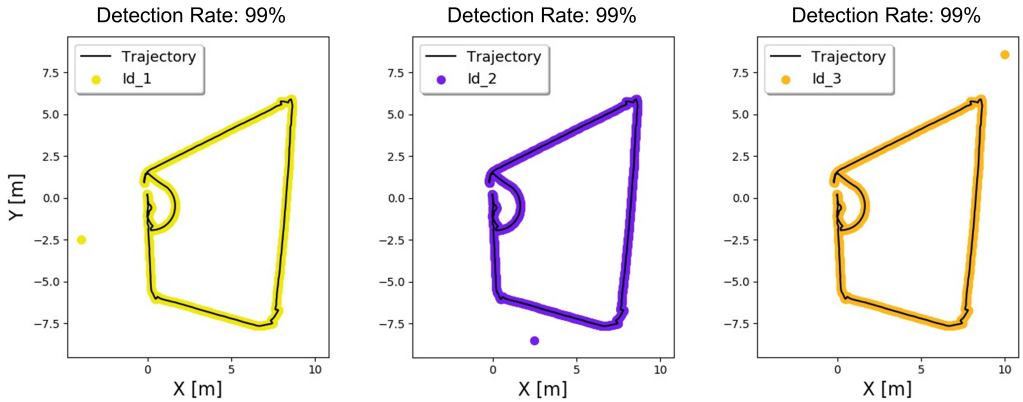


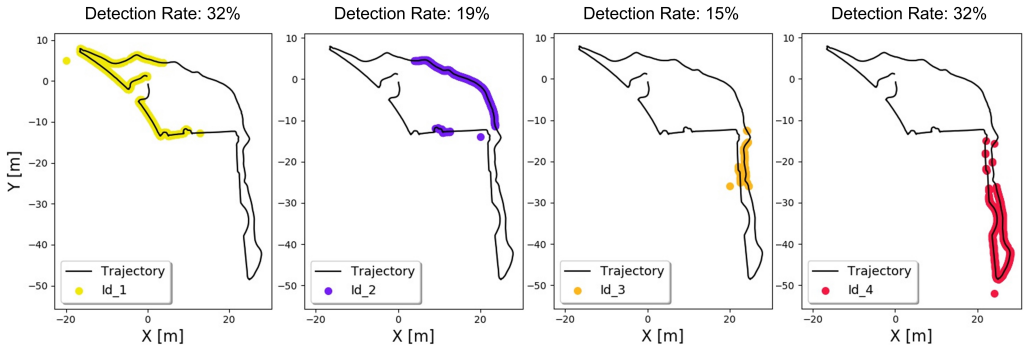
FIGURE 7.6: **Experimental Setup**: (a) shows the robot's traversed trajectory in small-scale and large-scale environments. (b) shows an example of a pre-trained anchor (i.e., a suitcase) and a derived landmark (i.e., an electrical cabinet) relative to the anchor in a real-world environment. (c) shows some scene-specific objects (e.g., hut) or regions (e.g., building corner, garden entrance) learned as natural landmarks.

Small-scale Garden

The training and test sets include images from a similar robot trajectory (fig. 7.6a) but in different environmental conditions (i.e., daytime, weather, and dynamic objects). The system used one recording to learn the scene representation and the other two to evaluate the localization performance. The training set consists of 1138 images, while the two test sets have 1091 and 1109 images. The first test set has more similar conditions w.r.t the training set, while the second set has more different illumination conditions and dynamic objects. The small garden consists of three learned landmarks, which the system used to perform landmark-based mapping and localization.



(a) Small Garden



(b) Big Garden

FIGURE 7.7: **Landmark Detection**: (a) and (b) visualize landmark detection in one of the recordings from small- and large-scale environments, respectively. The small garden has three landmarks, while the big garden contains four landmarks. Due to the smaller area, the learned landmarks are detectable in almost all images from the small garden. In contrast, most of the time, only a single landmark is detectable in a specific part of the big garden.

FIGURE 7.8: **Scene Variation due to Dynamic Objects**: Example images from the training set (a) and a test set (b) from the small and big gardens. Images show scene variation caused by a dynamic object. For the second example, the dynamic object mostly occludes the learned landmark in the scene.

After training PoseNet, it achieved a localization accuracy of 0.06m on the validation set. Table 7.2 shows the detection rates and the median localization accuracy w.r.t to the learned landmarks and the baseline methods. The use of a single landmark enables coarse localization in the environment. The localization accuracy of individual landmarks significantly drops for the second test set mainly due to the dynamic objects that were not present in the training set (fig. 7.8). Although the trained detector recognizes the landmarks with partial occlusions, scene variation affects the learned spatial representation and thus results in reduced localization performance. In the case of a full occluded landmark, the robot must rely on other landmarks in a scene. On the other hand, the combination of landmarks achieves similar or higher accuracy than the baseline methods on the test sets. Figure 7.10 visualizes the influence of adding more landmarks in a scene on localization for one of the test sets. Fourier-SFA and PoseNet produce good localization results when the environmental condition between the training and the test sets is almost identical (e.g., the first dataset from the small garden). However, the results degrade for the second test set.

Test	**Landmark-based Localization**					**Fourier-SFA**	**PoseNet**
Set	**Id_1**	**Id_2**	**Id_3**	**Combined**	**%**		
1	0.26m [99 %]	0.31m [99 %]	0.60m [99 %]	0.20m	100	0.19m	**0.18m**
2	0.73m [99 %]	0.93m [97 %]	1.33m [99 %]	**0.75m**	100	1.01m	0.83m

TABLE 7.2: **Real-world Small-scale Localization:** Median localization performance w.r.t learned landmarks and the baseline methods. The learned landmarks achieved nearly perfect detection rates, and no false positives were detected. The individual landmarks enable coarse scene localization while their combination performs similarly or better than the baseline methods. Please refer to the fig. 7.9 for the error distribution of both test sets for learned landmarks and baseline methods.

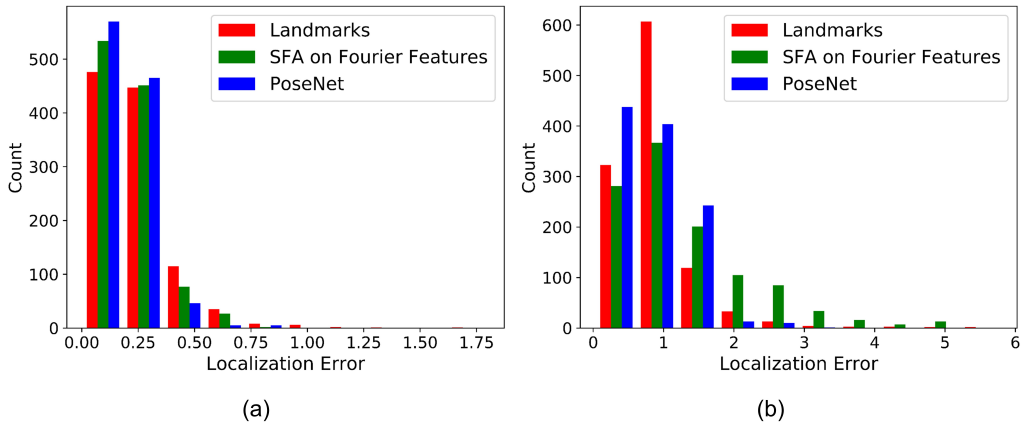


FIGURE 7.9: **Error Distribution (Small Garden)**: (a) and (b) show the error distribution of the two test sets for localization on learned landmarks (combined), Fourier-SFA, and PoseNet.

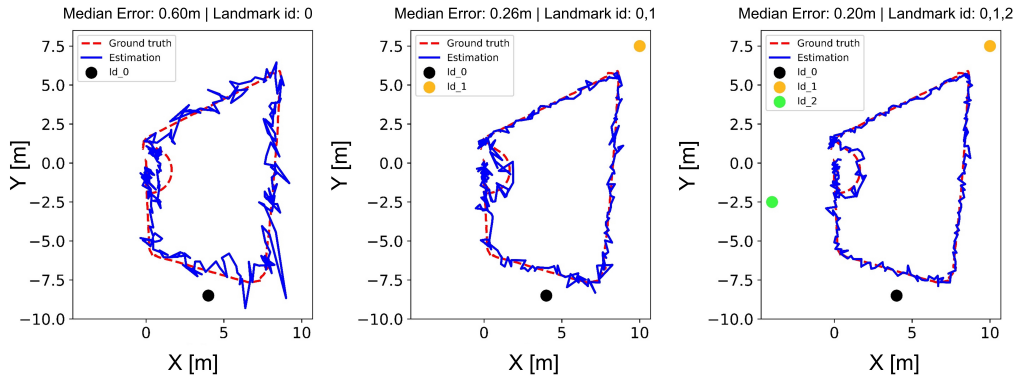


FIGURE 7.10: **Influence of Landmarks on Localization**: A human interactor iteratively adds new anchors to a scene. The robot learns the landmarks relative to the placed anchors for the localization task. Using a single landmark gives a rough estimate of the robot's position (left), while adding more landmarks provides the precise location estimation (right). Hence, the robot can cooperatively scale the system to achieve task-dependent localization accuracy.

Large-scale Garden

Like the small garden experiments, the system used one recording for learning the spatial representation and two separate sets for evaluation. However, one of the main differences is that this garden is much bigger and more complex than the small garden. Moreover, the learned landmarks are visible only in specific parts of the garden; thus, localization w.r.t only one landmark is possible for the most part. The number of training set images for the big garden is 4336, while the two test sets consist of 4032 and 4050 images. After training, PoseNet's localization accuracy on the validation data is 0.41m. Table 7.3 shows the localization results based on four learned landmarks, their combination, and the baseline methods. The results obtained with individual landmarks enable coarse localization in an environment. Nevertheless, their combination achieves better localization accuracy than the baseline methods. This effect, however, is more pronounced for the more challenging second test set from the big garden. Fourier-SFA does well on the datasets from the small garden. However, it does not scale to this environment. Similarly, PoseNet localization accuracy significantly drops on the test sets from the big garden. To conclude, the proposed landmark-based approach enables large-scale visual localization.

7.4.3 Scaling Experiments

This experiment aims to analyze the effect of increasing the number of landmarks on localization using simulated and real-world data from both gardens. The system used up to ten derived landmarks using the anchors present in a scene. The first step is

Test Set	Landmark-based Localization						Fourier-SFA	PoseNet
	Id_1	Id_2	Id_3	Id_4	Combined	%		
1	1.74m [32 %]	2.44m [16 %]	1.46m [13 %]	3.54m [25 %]	**2.22m**	78	7.50m	2.99m
2	2.06m [32 %]	2.97m [17 %]	1.83m [13 %]	3.80m [28 %]	**2.57m**	80	8.22m	6.57m

TABLE 7.3: **Real-world Large-scale Localization:** Median localization performance w.r.t learned landmarks and the baseline methods. The drop in the detection rates is due to the landmark visibility in a specific part of the scene. Please note that the system only used test images for the baseline methods, where at least a single landmark view was available in the corresponding image. Individual landmarks enable coarse localization while their combination outperforms the baseline methods. Please refer to the fig. 7.11 for the error distribution of both test sets for learned landmarks and baseline methods.

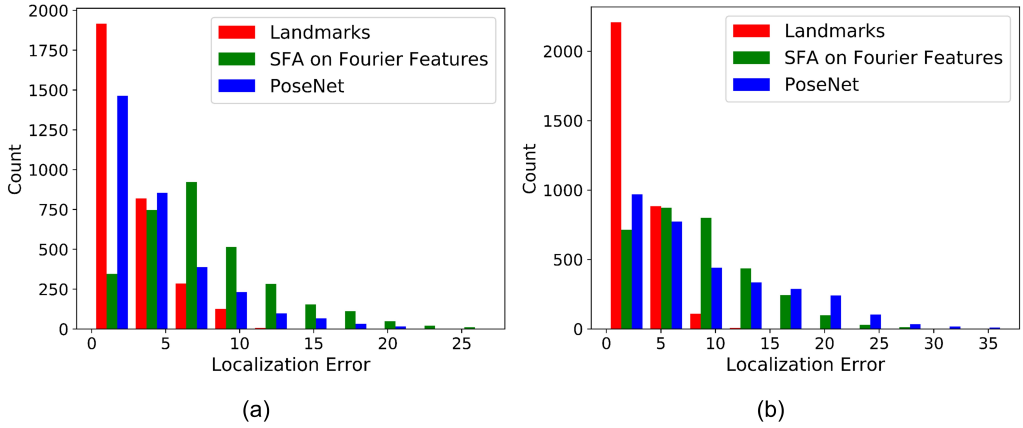
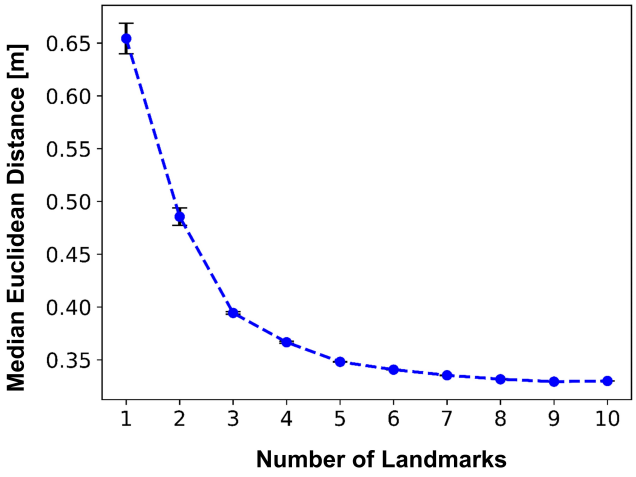
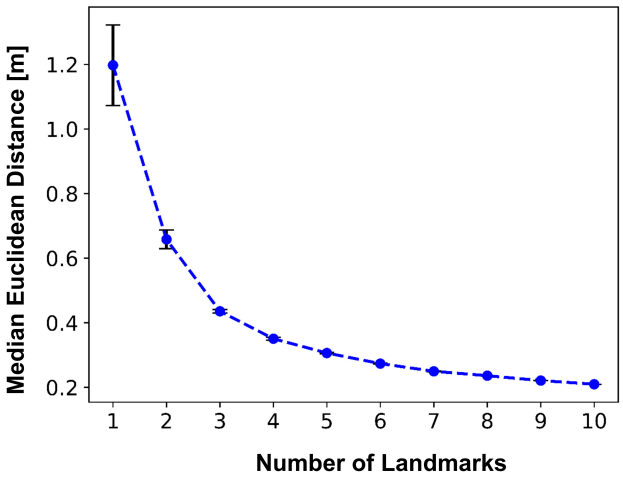


FIGURE 7.11: **Error Distribution (Big Garden)**: (a) and (b) show the error distribution of the two test sets for localization on learned landmarks (combined), Fourier-SFA, and PoseNet.

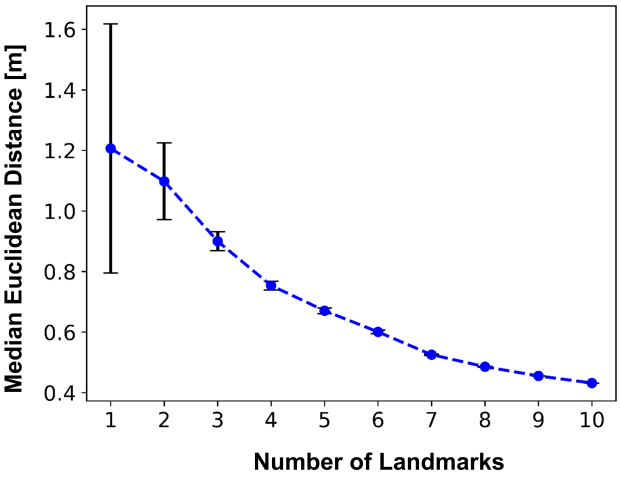
to learn an independent position estimator for each landmark. The second step processes landmark images from the test set using the estimators and predict the robot's 2D position (x, y). Afterwards, the final step calculates the test set's median localization error by systematically increasing the number of landmarks. Fig. 7.12 shows the results of 50 random permutations of the ten landmarks for both simulated and real-world data. The plots show an improved localization performance by increasing the number of landmarks until it saturates as expected. From an application perspective, a robot could increase the number of landmarks to achieve a certain accuracy level at runtime, depending on the area where high accuracy is required. A slight drawback is that since the system trains a separate SFA network for each landmark, the processing time will linearly increase (i.e., O(n)) with the number of landmarks. However, SFA-based mapping and localization are computationally inexpensive due to unsupervised learning. Hence, the system can scale from low computational cost and localization accuracy to higher accuracy at a modestly higher computational cost.



(a) Simulated data



(b) Small Garden



(c) Big Garden

FIGURE 7.12: **Scaling Experiments**: Simulated data (a). Real-world data (b,c). The plot shows the median and variance localization accuracy for 50 random permutations of the ten landmarks. Using more landmarks for localization improves the accuracy initially, and eventually, it saturates for a higher number of landmarks.

7.5 Discussion

This chapter introduced a novel approach to speed up the label generation process for learning new visual landmarks in a scene. The method uses physical instances of pre-trained CNN objects as anchors to generate labeled data for the unseen imagery. Pre-trained detectors like MS-COCO are mainly available for everyday manufactured mobile objects, which makes them unsuitable as landmarks. The proposed approach turns this disadvantage into an advantage since typical mobile objects (e.g., a bicycle) can be easily acquired and placed temporarily as anchors. Therefore, a human only needs to place an anchor in the scene and specify its spatial relationship to a new landmark *once* instead of potentially hand-labeling thousands of images. Please note that a human is not strictly required to suggest new landmarks but to provide a temporarily stable and unique training anchor object. However, a human's common sense knowledge can help select landmarks that will be stable and unique over time (e.g., manufactured objects rather than vegetation to not change too much with changing seasons). Afterwards, the system automatically generates the labeled data and trains a detector to learn the landmarks. Most machine-learning approaches work on pre-recorded labeled image data, whereas the proposed approach requires physical interaction on-site with the robot.

After the landmark learning phase, the system used landmark views to perform localization experiments. The results show that landmark-based localization significantly outperforms the baseline methods in a large-scale environment and achieves similar results in a small-scale environment. In the large-scale environment, scene variations during training affect the learning of spatial representation with SFA on complete images. However, focusing on smaller image parts (i.e., landmarks) helps to reduce such effects and thus achieve higher localization accuracy. The occlusions produced by dynamic objects affect the localization performance based on individual landmarks. However, the method's performance can be further improved by integrating more landmarks and obtaining a combined position estimation relative to them. A difference from classic landmark-based localization approaches, for example, triangulation, is that the proposed method enables localization from a single landmark. In contrast, triangulation requires the simultaneous detection of at least three landmarks with a known position. The landmark-based localization here uses SFA as an efficient unsupervised feature learning step. However, the cooperative landmark learning approach may also improve standard supervised pose regression methods like PoseNet.

The labeling approach allows learning actual semantic objects as new landmarks. When anchor and landmark are within the same plane, perspective changes during

recording result in labeling the identical scene part as a landmark in the training paradigm with a fixed 2D offset between anchor and landmark. In the most extreme case, if an anchor is placed such that the robot can go around it, a simple 2D approach may fail and not capture a semantically meaningful region but a subset of the scene's viewing space. Geometry-based localization methods may fail in such a case. However, unintuitively, these views are classified very well as a landmark with a CNN and thus have no influence on localization accuracy obtained with position regression learning (as shown here with SFA). Hence, the new landmark may not necessarily contain complete objects; it can be a non-object patch due to perspective effects. As long as the patch is informative (e.g., not a part of the sky), the results show that localization learning performs well on such landmarks. From an application perspective, the system is suitable for service robots (e.g., lawnmowers and vacuum cleaners), employing a pre-trained visual detector to learn new landmarks in a scene. Thus, the approach enables reliable localization in the long term, even if the anchor objects are no longer present in the scene. In summary, the following are the main advantages of the methods introduced in this chapter:

1. The proposed landmark-learning approach enables efficient and fast generation of labeled training data that only requires minimal human intervention in contrast to tedious hand-labeling.
2. Localization relative to learned landmarks in a scene allows large-scale localization, an essential aspect of any localization approach.
3. Incorporating multiple landmarks improves the localization accuracy and thus allows to adjust it according to the needs.

However, a slight disadvantage is that the introduction of landmark learning in the SFA-based localization pipeline makes the approach computationally more expensive than SFA on raw pixels or Fourier Features since it requires a GPU to learn and detect the landmarks in images. The following section presents an alternative approach for performing localization independent of any pre-fixed visual landmarks.

7.6 Visual Localization using Generic Landmarks

The idea is to remove the dependency on learning pre-fixed visual landmarks for localization by implementing a generic system that learns to localize from an image patch, i.e., a small group of nearby image pixels (fig. 7.13). To this end, the system uses SFA to encode the robot's position (x, y) as the slowest feature in the learned representation. Here, the proposal is to exploit the slowness property of SFA to

encode the robot's gaze point in the scene as the slowest feature by changing the training data statistics. The resulting representation will smoothly encode the spatial relationship of gaze points and, consequently, be used to regress a robot's 2D-position (x, y) from an image patch without costly deep learning.

FIGURE 7.13: **Generic Landmark-based Localization**: The aim of anywhere localization is to use any image patch to estimate the robot's position (x, y) instead of pre-fixed landmarks in a scene. Just like landmarks, these individual patches (i.e., generic landmarks) will give an independent estimate of the robot's position (x, y).

7.6.1 Proposed Method

The core idea here is to encode a robot's gaze as a slowly varying feature by changing the input statistics of the training data. Figure 7.14 shows the procedure of generating a training sequence in which a robot's gaze changes slowly over time compared to other variables, for instance, a robot's position (x, y). The robot fixates its viewing direction on a region in the scene. It now starts traversing the scene while focusing on the fixated region for the complete trajectory. After returning to the starting position (x, y), it shifts its gaze to a neighboring region in the scene and continues traversing. The robot repeats this process until ideally whole image space has been considered. The application of SFA on such data will smoothly encode the spatial relationship between the gaze points as nearby regions will get similar slow feature values due to SFA's slowness objective. The learned representation can be later projected to metric space (x, y) by learning a regression function similar to the previous approaches. Hence, the approach estimates the 2D robot position (x, y) from any image patch, irrespective of any learned landmarks with a CNN.

From the implementation point of view, the desired sequence can be easily generated by incorporating a visual tracker in the SFA localization pipeline. Generally, the tracking algorithms are faster than the detection algorithms, which makes them suitable for many real-world applications. Hence, the straightforward approach is to use an available visual tracker from OpenCV [3] to generate the desired training sequence, as mentioned earlier. However, some challenges prevented the use of such trackers in this work. The primary reason is that the OpenCV trackers drift a lot as the robot moves, making it harder to generate the desired training sequence for encoding the

[3] https://docs.opencv.org/

robot gaze. In addition, the application environment consists of many similar things (i.e., brick walls and bushes). Such trackers latch on to nearby regions and do not return to their initial location in the image. All these issues make it challenging to generate the desired training sequence. Since the development of a reliable tracker is beyond the scope of this thesis, the following section presents the experimental results from the prototype implementation of the proposed method, where the aim is to simulate a perfect tracker and analyze the localization results.

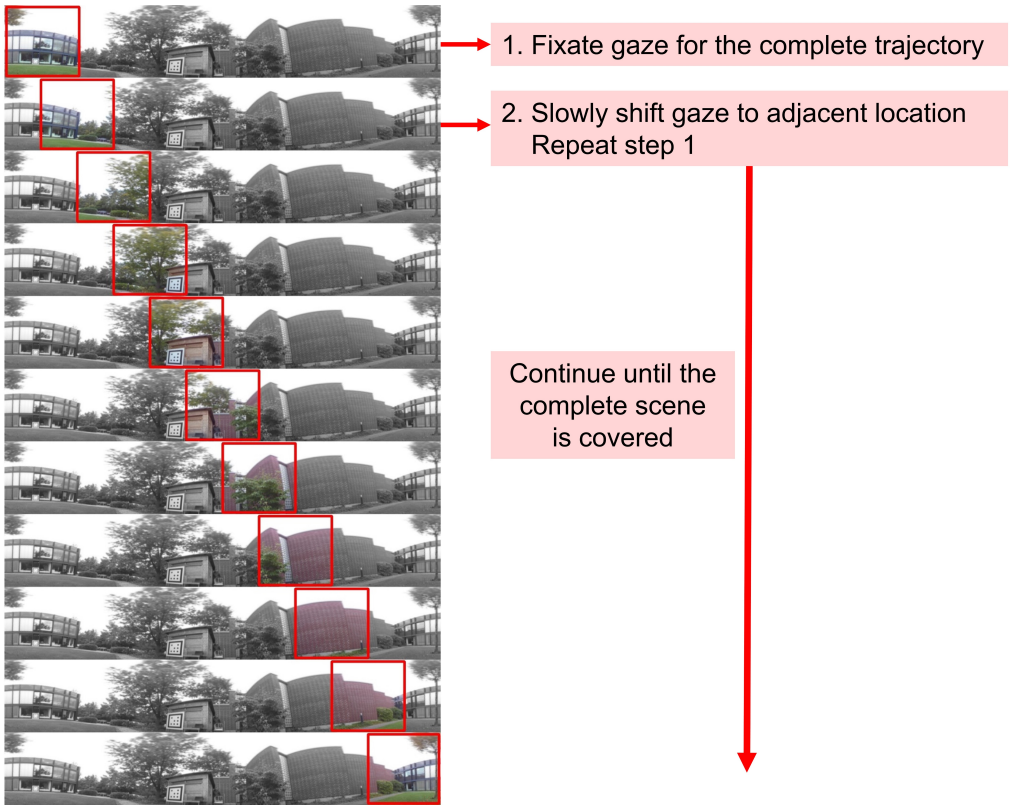


FIGURE 7.14: **Training Sequence Generation for Encoding Robot's Gaze**: The robot starts with tracking a fixated region for some time (here: for a fixed trajectory). Afterwards, it switches the viewing region to the adjacent location and continues traversal. It repeats the procedures to cover the entire scene. The application of SFA on the resulting training sequence should ideally encode the robot's gaze as the slowest feature in the learned representation.

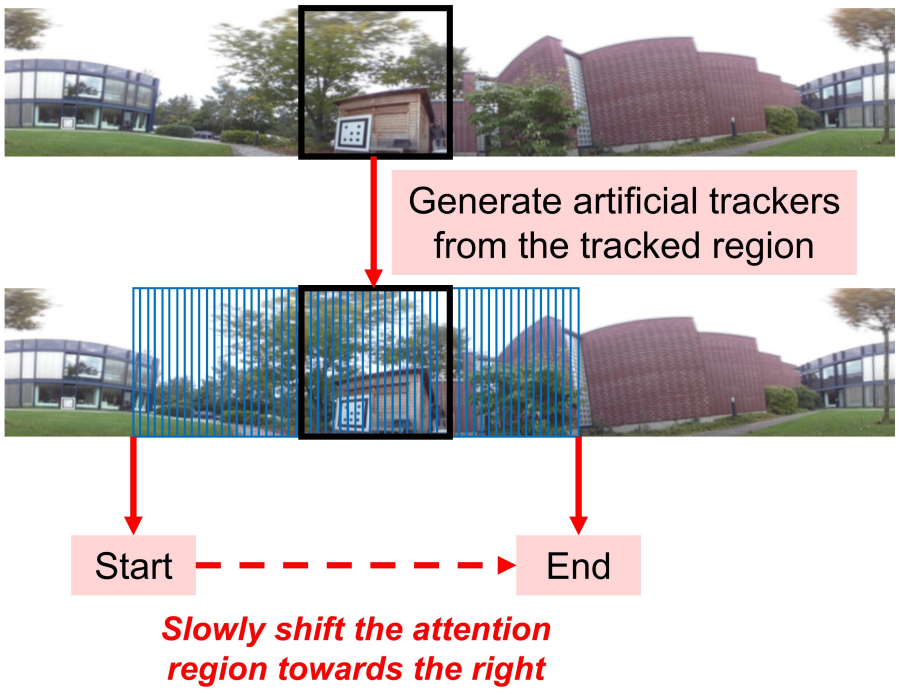
7.6.2 Experimental Results

The prototype version of the approach tracks a single region in the scene and generates artificial trackers relative to it using a simple 1D offset. Figure 7.15a shows the tracked region in the image sequence and the generated trackers from its position in the image. For the proof of concept, the system did not track the entire scene; instead, it uses a subset of the scene's viewing space. Afterwards, the system starts with the left-most tracker (as indicated in the figure) and shifts the attention region

towards the right to generate the desired training data. Later, this data is fed to a hierarchical SFA network to encode the spatial relationship between the robot's gaze points. After learning the representation, the system projects the learned SFA space representation to the metric space by learning a regression function. During the test phase, the system randomly takes an image region (fig. 7.15b) and estimates the robot's position (x,y) using it. Please note that the figure only shows a single region. However, it is possible to use multiple regions at the test time and combine their estimation for higher localization accuracy. Figure 7.16 shows the localization results obtained from the proposed approach and learned landmarks as the baseline method. The median localization accuracy using three generic landmarks is 0.17m, while the accuracy is 0.20m using the three learned landmarks. Thus, the results demonstrate that the proposed method is a viable alternative to learning pre-fixed landmarks for localization. The future direction would be to investigate incorporating advanced context-aware trackers (Mueller et al., 2017) in the pipeline and generate the training data as discussed.

7.7 Conclusion

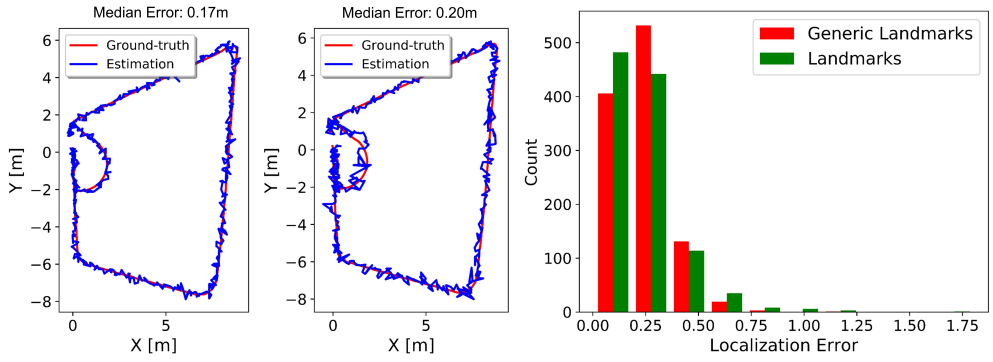
This chapter introduced a new SFA-localization approach incorporating visual landmarks into the localization pipeline. Compared to the cumbersome hand-labeling method, the *cooperative landmark learning* approach allows efficient and fast generation of labeled training data for learning new landmarks in a scene. The experimental results show that the learned landmarks are well-suited for localization and achieve higher accuracy than the baseline methods in a large-scale environment. Moreover, combining multiple landmarks for localization further improves the accuracy. Hence, the system enables scaling the localization accuracy according to the needs. The chapter also presented an alternative approach that encodes the robot's gaze instead of its position in the learned representation as slowly varying features. This approach allows a robot to localize from any image patch (i.e., generic landmarks) in contrast to learning pre-fixed landmarks with a CNN. However, due to the visual tracking issues, the approach is currently limited to its prototype version. The prototype implementation shows promising localization results. Hence, the future work is to use some stable tracker and learn gaze representation for the entire scene, allowing a robot to localize from looking *anywhere* in the scene.



(a) Training Phase

(b) Testing Phase

FIGURE 7.15: **Prototype Implementation**: (a) shows the steps to generate training data for implementing the prototype version of the approach. The system tracks a single region using an OpenCV tracker and drives artificial trackers relative to the tracked area. The system then extracts the tracked regions from left to the right and feeds them to the SFA network for training. (b) The localization phase uses a random region (here: from a subset of the scene's viewing space) and estimates the robot's location (x, y).



(a) Generic Landmarks (b) Learned Landmarks (c) Error Distribution

FIGURE 7.16: **Localization Result**: (a) and (b) show the estimated trajectory using three generic and learned landmarks. (c) shows the error distribution for both approaches.

Chapter 8

Fast Visual Localization and Mapping with Slow Features

This chapter is based on the following peer-reviewed publication:

- Haris, M., Franzius, M., & Bauer-Wersing, U. (2021). Unsupervised Fast Visual Localization and Mapping with Slow Features. In IEEE International Conference on Image Processing (ICIP, pp. 519-523). IEEE

Visual localization is the task of accurately estimating the camera's position in a known environment. State-of-the-art structure-based methods (Sattler et al., 2012; Sattler et al., 2017) are well-known for precise, fast, and generalizable visual localization approaches. They are based on the laws of projective geometry and the underlying 3D structure of a scene. In contrast, training convolution neural networks (CNNs) for visual localization has recently gained significant interest. These methods (e.g., Kendall et al., 2017) train a neural network in an end-to-end way to directly predict the pose of an image. This chapter systematically compares SFA-based localization with *Active Search* (i.e., a structure-based method) and *PoseNet* (i.e., an end-to-end learning-based approach) in real-world test scenarios w.r.t localization accuracy and computation time.

Active Search (Sattler et al., 2012) is one of the fastest and most widely used structure-based localization methods, which estimates the pose of a test image relative to a pre-computed 3D scene reconstruction. It proposes a faster pipeline that combines 2D-to-3D and 3D-to-2D search into an active correspondence search step, which allows the method to achieve similar or higher registration performance compared to tree-based search and results in faster run times. Moreover, it achieves comparable or superior localization performance in small- and large-scale environments compared to various localization methods (Sattler et al., 2017; Sattler et al., 2019). Similar to SFA-based localization, both Active Search and PoseNet have non-simultaneous mapping and localization phases, making it possible to compare the methods in a

straightforward and meaningful way.
The experiments first test the methods on subsequent robot recordings along the same path (i.e., temporal generalization) and then for the case when the train and test recordings consist of sufficiently different robot trajectories (i.e., spatial generalization). The performance metrics are localization accuracy and computation time measured separately for mapping (training) and localization (test) phases. In summary, this chapter's motivation is two-fold: First, to show through real-world experiments if an unsupervised learning approach can compete with the baseline approaches w.r.t localization accuracy. Second, to compare the methods, w.r.t run-time, and hardware requirements in the mapping and localization phase, respectively.

8.1 Structure-based Localization

Active Search determines the pose of a test image relative to an offline built 3D-point cloud reconstruction of a scene. The available implementation[1] of Active Search expects the input (3D reconstruction) in the Bundler[2] file format. However, Bundler uses a relatively simple pinhole camera model and cannot handle more sophisticated camera models, for instance, fisheye lenses. Due to this limitation, the system reduced the field of view (FoV) of fisheye images to 90° and computed the 3D scene reconstruction with COLMAP's (Schönberger et al., 2016) pinhole camera model. Despite the reduced FoV, the remaining image content was sufficient to generate an excellent 3D point cloud of the scene with the proposed approach (fig. 8.1). Afterwards, the system registered the 3D model against the odometry information obtained during the recording to recover the model's scale. The final step was to export the COLMAP's output based on a pinhole model to the Bundler format as required by Active Search. In the localization phase, the system also reduces the FoV of test images to 90°. Please note that the system did not use test images while computing the 3D scene reconstruction to prevent their influence on the final model. The localization phase uses the 3D model and the Active Search pipeline (Sattler et al., 2012) to estimate the pose of the test images.

8.2 Experiments

The experiments were performed using the datasets collected from an outdoor environment with an area of 15×9 meters. An autonomous robot traverses the area's border (i.e., first working phase) and then traverses freely (i.e., second working phase) to

[1] https://github.com/hanjianwei/ACG-Localizer
[2] http://www.cs.cornell.edu/ snavely/bundler/

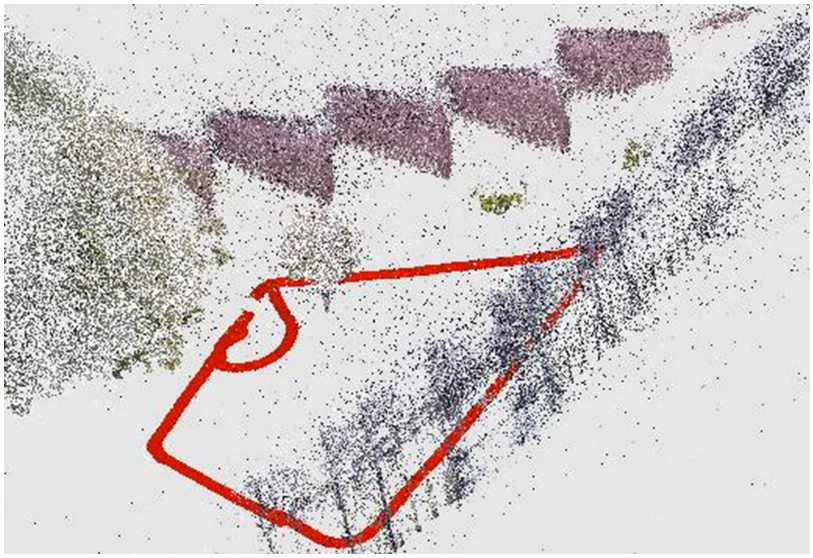


FIGURE 8.1: **3D Scene Reconstruction**: A sparse scene reconstruction was computed with COLMAP's structure-from-motion (SfM) pipeline (Schönberger et al., 2016). The image dataset contains an ordered sequence of 1083 images collected by a fisheye lens mounted on a robot.

store scene images. The robot captures omnidirectional images of size 2880×2880 pixels. Due to the issue mentioned earlier regarding the baseline method, an initial pre-processing step reduces the field of view (FoV) of all collected images to 90°. The size of each image with the reduced FoV is 1440×1440 pixels. Please note that this additional pre-processing step is only done for the experiments presented in this chapter. Figure 8.2 shows the robot's traversed trajectories during a recording session and some example images with reduced FoV.

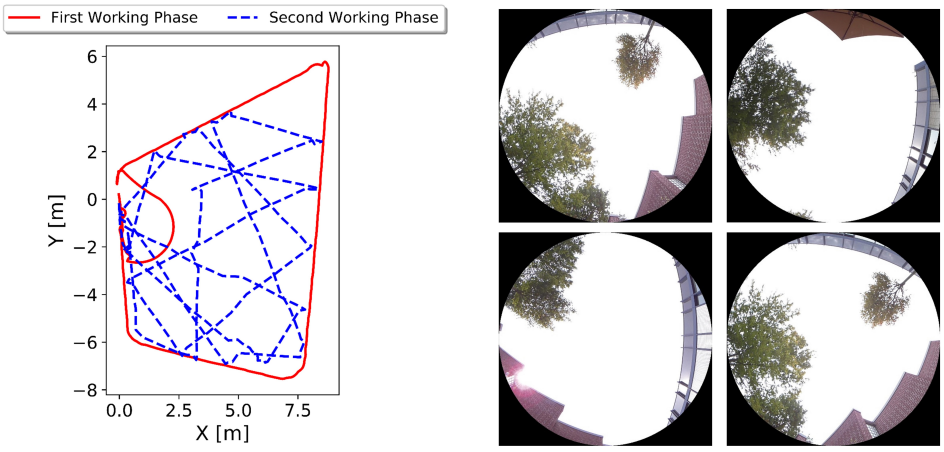


FIGURE 8.2: **Experimental Setup**: (a) First working period (red): robot follows the border wire; second working period (blue): robot traverses freely within the area enclosed by the wire. (b) shows some example images from the data sets used for the experiments.

For the SFA part, the system projects the input images with dimensions 1440×1440 to panoramic views of size 600×60. The later steps involve Fourier feature extraction from the views and using these features to learn SFA representation. After the training phase, the next step uses the trained model and the training data to compute the eight slowest features $s_{1\ldots8}$ for each position (x, y). The system then uses the learned slow features and trains a separate regression function for x and y metric ground-truth positions to evaluate metric performance. Similarly, the final step computes the test data's slow features and uses the learned regression function to predict test set locations (x', y'). The procedure for obtaining 3D scene reconstruction for Active Search has already been described in section 8.1. During the localization phase, the system extracts SIFT features (Lowe, 2004) of the test images using COLMAP[3] and runs the Active Search pipeline to estimate the pose for each test image. For obtaining PoseNet results, the system unwraps 90° omnidirectional views to 1200×200 pixels and then learns to regress the image pose by training a neural network in an end-to-end fashion (Kendall et al., 2017). For all the methods, the system computes Euclidean distance between estimated locations (x', y') and the ground-truth test set locations (x, y). The experiments include testing localization methods in two different scenarios, i.e., short-term temporal generalization and spatial generalization w.r.t localization accuracy and computation time.

8.3 Results

8.3.1 Short-term Temporal Generalization

This experiment aims to test the re-localization ability of the methods over a short time. The training and test set images were recorded on a similar trajectory but over a time difference of 30 minutes, causing a slight variation in lighting conditions. The training and test sets consist of 1083 and 284 images, respectively. Table 8.1 reports both median and mean localization accuracy, and fig. 8.3 visualizes the localization error for the methods. Active Search performed better in this experiment than SFA-based and PoseNet localization.

8.3.2 Spatial Generalization

This experiment aims to test the methods' re-localization ability when the train and test sets contain images from significantly different robot trajectories but in similar

[3] https://colmap.github.io/

Method	Median [m]	Mean [m]	Quartiles [m]	
			3rd	4th
SFA	0.48m	0.63m	0.78m	5.03m
Active Search	**0.28m**	**0.42m**	0.51m	5.82m
PoseNet	0.42m	0.66m	0.93m	3.79m

TABLE 8.1: **Localization Results for Temporal Generalization**: Median and Mean Euclidean error of the localization methods for generalization over time. Active Search outperformed both SFA and PoseNet localization in this experiment. For detailed information on error distribution, please refer to fig. 8.4.

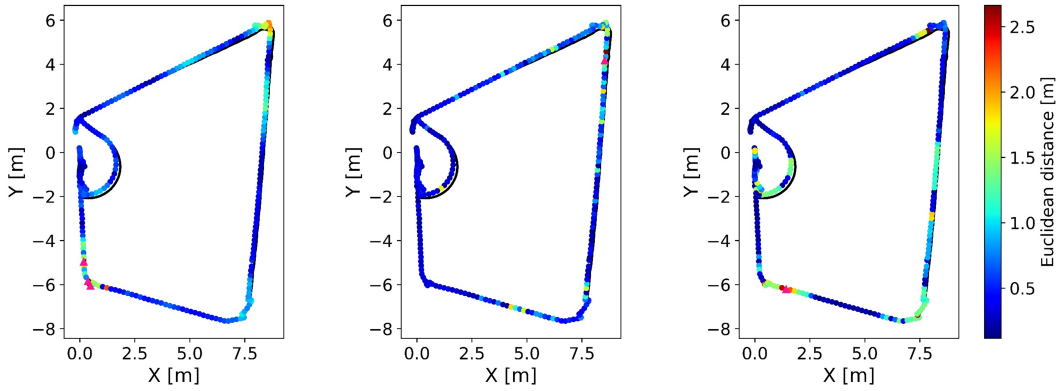


FIGURE 8.3: **Visualization of Localization Results (Temporal Generalization)**: The robot follows a similar trajectory (border wire) twice to collect images for the training and test set. The time difference between the two recordings is 30 minutes. (a-c) visualizes the localization error as indicated by the color codes for all the methods. Pink triangles indicate the positions where the estimated error is higher than three meters. Active Search has achieved better median and mean localization accuracy than other methods.

conditions. Therefore, the system uses images captured by the robot from the border and inner field positions as training and test data (fig. 8.2a). The training and test sets consist of 1141 and 298 images, respectively. The ground-truth data (x, y) for the open field was obtained using commercial photogrammetry software, i.e., Metashape, as discussed in chapter 4. Table 8.2 reports both median and mean localization accuracy, and fig. 8.5 visualizes the localization error of the methods. Active Search has achieved better median localization accuracy than SFA and PoseNet localization. However, when comparing the mean accuracy, SFA has shown better results. The mean localization error for SFA is 1.04m, while the error is 2.52m and 1.95m for Active Search and PoseNet, respectively. It is because almost 20% of the position estimates (out of $n = 298$) produced by Active Search and PoseNet have localization errors greater than three meters, especially in the center of the traversed area (c.f.

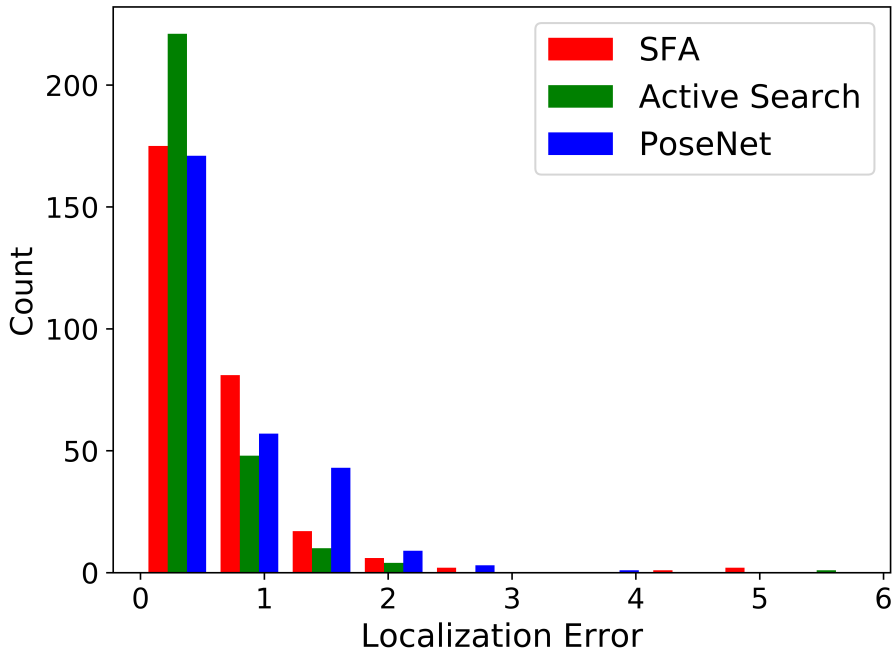


FIGURE 8.4: **Error Distribution (Temporal Generalization)**: It shows the error distribution $(n = 284)$ of the test set for localization using SFA, Active Search, and PoseNet.

fig. 8.5). Moreover, Active search is prone to outliers and thus produces some extreme ones (c.f. tab. 8.2). On the other hand, only 3% of estimates have localization errors greater than three meters for SFA-based approach. Thus, the SFA localization produced slightly more robust results than the baseline methods in the case of spatial generalization. This performance can be significantly improved by adding some sparse images to the training sequence from the infield locations. Please note that for SFA, this step is highly feasible since it does not require any labeled data to learn scene representation in contrast to any end-to-end learning-based approach like PoseNet. Moreover, the results obtained with PoseNet for this experiment also align with the work (Sattler et al., 2019), where the authors show that end-to-end learning does not necessarily generalize beyond the training set data.

Method	**Median [m]**	**Mean [m]**	**Quartiles [m]**	
			3rd	**4th**
SFA	0.72m	**1.04m**	1.41m	10.37m
Active Search	**0.38m**	2.52m	1.81m	94.65m
PoseNet	1.65m	1.95m	2.82m	7.32m

TABLE 8.2: **Localization Results for Spatial Generalization**: Median and Mean Euclidean error of the localization methods for generalization over space. For detailed information on error distribution, please refer to fig. 8.6.

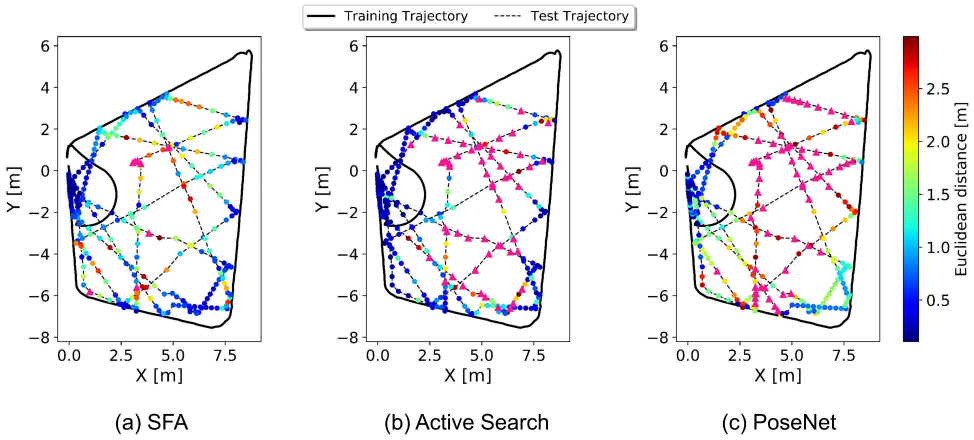


FIGURE 8.5: **Visualization of Localization Results (Spatial Generalization)**: The robot follows the border wire (solid line) to collect the training set images and freely traverses within the border wire area (dashed line) to collect test set images. (a-c) visualizes the localization error of the methods as indicated by the color codes. SFA performs well in spatial generalization and moderately degrades for test images towards the center of the field. Active Search has achieved better median localization accuracy than SFA and PoseNet localization, but it is worse in the mean performance. As expected, Active Search has quite good estimations near the border wire, but it degrades for the test images from the center of the field. Pink triangles indicate the positions where the estimated error is higher than three meters. PoseNet achieves a moderate localization accuracy near the border and becomes worse for the center of the field.

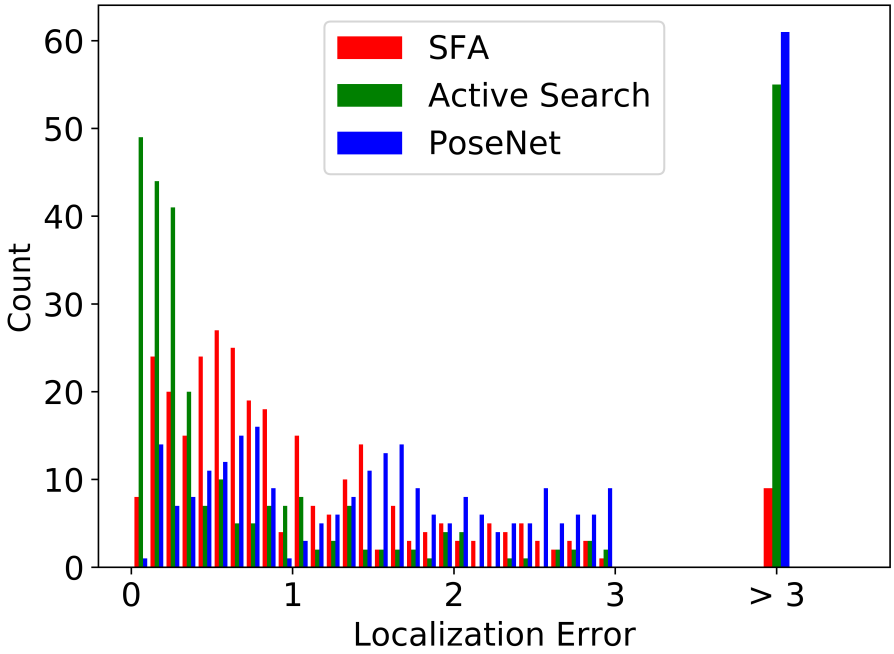


FIGURE 8.6: **Error Distribution (Spatial Generalization)**: It shows error distribution $(n = 298)$ for the methods. Almost 20% of the test locations predicted by Active Search and PoseNet have more than a three-meter error. In contrast, for SFA, only 3% of locations have an estimated error greater than three meters. Thus, SFA shows a better tendency to generalize to unvisited positions.

8.3.3 Time Evaluation

The time evaluation was performed separately for the mapping and localization phases. The system uses the dataset containing 1083 training and 284 test images. Both phases of the SFA-based method were computed on a standard CPU (Intel i5 – 6500 with 3.20GHz). For Active Search, the compute hardware includes a GPU (GeForce RTX 2070) for 3D scene reconstruction (mapping) and a standard CPU (as mentioned earlier) for localization. In contrast, both phases of PoseNet were computed on a GPU (GeForce RTX 2070). The mapping phase in the table 8.3 reports the preprocessing and mapping times for generating scene representation using the training set images. The pre-processing time for SFA indicates the time required to project fisheye images to panoramic views and extract Fourier features. The corresponding time for 3D scene reconstruction shows the time to extract SIFTGPU features (using GeForce RTX 2070) from the fisheye views. For PostNet, this time indicates the projection of omnidirectional to panoramic views. The mapping time for SFA includes the time required for SFA training and obtaining metric representation. The mapping time for scene reconstruction time indicates the time to create the 3D structure of the scene. The corresponding time for PoseNet shows the time to train a neural network to directly regress a pose from an image. The localization phase in the table 8.3 presents the time it took to localize 284 test images for all methods. The pre-processing time for Active Search includes the time to extract SIFTGPU features from the test images and to extract the relevant information from the created 3D model as needed by Active Search. Based on the results, the SFA-based method for this dataset is 886x and 370x faster in mapping than 3D scene reconstruction and training a neural network, respectively. Similarly, it is 34x and 3x faster than Active Search and PoseNet in the localization phase. This factor could be further increased by optimizing the preprocessing for SFA, which is currently implemented in python and requires more than 85% of the overall localization time.

8.4 Conclusion

The experiments in this chapter aimed to compare the unsupervised SFA-based approach with structure-based Active Search and learning-based PoseNet methods for visual localization in a real-world setting. The chapter reported the localization accuracy and computation times for the training and testing phases as performance metrics. Although Active Search has achieved better median localization accuracy in both experiments, some results are highly inaccurate (see Fig. 8.5b). This behavior might not be suitable for a robot operating in real-world scenarios. On the other

Phase	Method	Preprocessing Time [s]	Mapping Time [s]	Localization Time [s]	Total Time [s]
Mapping	SFA [CPU]	**44**	**5**	**-**	**49**
	3D Reconstruction [GPU]	90	43334	**-**	43424
	PoseNet [GPU]	120	18000	**-**	18120
Localization	SFA [CPU]	**12**	**-**	**2**	**14**
	Active Search [CPU]	185	**-**	294	479
	PoseNet [GPU]	32	**-**	10	42

TABLE 8.3: **Computation Time Evaluation**: The computation times in mapping indicate generating scene representation with the respective methods using 1083 training set images. The localization phase shows the time to localize 284 test set images. Based on the results, the SFA-based method for this dataset is **886x** and **370x** faster in the mapping phase and **34x** and **3x** faster in the localization phase than Active Search and PoseNet, respectively.

hand, the SFA-based approach achieves comparable performance to Active Search near the border wire but generalizes well to the infield locations. The method offers extremely fast run times for mapping and localization phases, with only slight degradation of accuracy w.r.t Active Search. However, both Active Search and PoseNet can provide a full 6D pose of an image in contrast to SFA, which offers a 2D position (x, y). Still, many service robots, such as lawnmowers and vacuum cleaners, do not require a 6D pose estimation for localization.

The drastically faster run times make the SFA-based localization an attractive solution to run on low-cost embedded hardware (e.g., on a lawn mower robot). Preliminary results show a further 30% improvement in localization performance using a Kalman filter to combine odometry with visual localization. To conclude, the experiments show the ability of SFA localization to generalize over time and space with a straightforward model. SFA-based localization generalizes better to previously unseen trajectories when compared to Active Search and PoseNet. However, the advantage that stands out is its much higher speed without needing a GPU.

Chapter 9

Live Navigation on a Service Robot

This chapter is based on the following peer-reviewed publication:

- Haris, M., Franzius, M., & Bauer-Wersing, U. (2018). Robot navigation on slow feature gradients. In International Conference on Neural Information Processing (ICONIP, pp. 143-154). Springer, Cham.

9.1 Introduction

One of the essential abilities of autonomous robots is navigation in space. For non-trivial navigation, a robot needs an internal representation of the environment to estimate its location and plan a viable path to a target. Visual simultaneous localization and mapping (vSLAM) enable a robot to build a map of the environment and estimate its location simultaneously using vision as the only sensory input. The resulting maps represent the environment in different ways, such as a graph structure representing the topology or a discretized occupancy grid, which leads to different navigation strategies (Fuentes-Pacheco et al., 2015). These strategies have different levels of complexity ranging from reactive motion execution to path planning in metrical maps (Meyer et al., 2003).
Navigation is a challenging task for mobile robots. Many animals, on the other hand, have excellent navigation capabilities. They may take suboptimal paths while reaching the target; however, the paths are flexible and quickly planned, which results in an adaptive and robust navigation behavior. One such model of rat navigation is RatSLAM (Milford et al., 2004). The pose is encoded by an activity packet in a 3D continuous attractor network with the axis representing (x, y, θ). Self-motion cues and visual template matching inject energy into the network, shifting the peak of activity. An extension of RatSLAM is the organization of unique combinations of local views and pose codes in a graph-like experience map, which enables the model to maintain a consistent spatial representation over extended periods (Milford et al.,

2010). Another class of models is based on slowness learning (Wyss et al., 2006; Franzius et al., 2007) and also focuses on localization as a model of the rodent hippocampus. In slowness learning (Wiskott et al., 2002), the resulting representation can achieve significant invariances, e.g., to head orientation. Using an uncalibrated omnidirectional imaging system, an earlier model was successfully applied to a mobile robot in an outdoor environment (Metka et al., 2013). The learned representations of such models can be projected into metric space for navigation. While such a step makes quantitative evaluation easier and allows, for example, the integration of Kalman Filtering, it requires additional computational effort and removes interesting information from the training phase.

Navigation in topological maps is relatively straightforward and can be performed using the graph search algorithm A* (Hart et al., 1968). Given an admissible distance heuristic, it is guaranteed to find the optimal path, but it is a memory and computationally intensive task for large environments with many obstacles. The potential field method is a different approach for navigation in metric space that is based on gradient descent in a vector force field defined by an attractor at the target position and repulsive forces from obstacles (Khatib, 1985; Barraquand et al., 1991). Although it is an elegant solution, a known limitation of the approach is local minima caused by certain obstacles or their spatial configuration (Tilove, 1990).

Reinforcement learning has been applied to the low dimensional representation of the environment, extracted from visual input using Slow Feature Analysis (SFA), to learn policies that guide an agent to a goal location in a simplified version of the Morris water maze task (Legenstein et al., 2010) and with views from a mobile robot (Böhmer et al., 2013). The presented results demonstrate the approach's feasibility but require an additional learning step for new goal locations.

Recent work (Metka et al., 2017) proposed an elegant approach for navigation in slow feature space using gradient descent. After the unsupervised learning of the environmental representation, navigation can be performed efficiently by following the SFA-gradient, approximated from distance measurements between the target and the current value. However, the work was limited to simulation and required multiple cameras to estimate the slow feature gradients. The approach presented in this chapter extends the previous work to become practical in several ways. Firstly, the system replaces the part of the slow feature hierarchy with a Fourier feature extraction as pre-processing step to improve robustness and computation speed and achieve invariance to the robot's orientation. Secondly, the approach estimates the navigation gradient using a single camera. Finally, the chapter presents quantitative results for navigation with a lawn mower robot in free space and around obstacles.

9.2 Learning Spatial Representations with SFA

In order to find a representation that is suitable for the navigation task, the goal is to learn functions that encode the robot's position (x, y) in space and are invariant to its orientation. While the objective of SFA enforces temporal slowness on the training trajectory, a suitable intermeshed training trajectory leads to representations that also change smoothly in space (Franzius et al., 2007). Furthermore, suppose the environment is suitably rich (i.e., without an identical appearance at different locations) and the function space for training is sufficiently large. In that case, each position in space is uniquely encoded by its Slow Feature representation. These properties allow a robot to navigate by following the gradient of the difference between the spatial representations of the current position and a target position. The kind of spatial information encoded in the learned slow feature representations depends solely on the statistics of the training data. If the robot's orientation changes on a faster timescale than its position, the slowest features become orientation invariant (Franzius et al., 2007). Earlier work simulated additional rotation by shifting a sliding window over the periodic panoramic images from a 360-degree camera on the robot (Metka et al., 2013; Metka et al., 2017). Learning orientation invariance, however, generates computational load during training, and the results are not perfect for limited training data. Noise on Slow Feature representations deteriorates the navigation gradient and is thus more problematic for gradient-based navigation than for localization. Consequently, the current system explicitly removes orientation dependency from the input representations for SFA with a pre-processing step. For this purpose, the system extracts row-wise Fourier Features and retains the magnitude part corresponding to the lowest 15 components (c.f. chapter 5). It provides a straightforward to achieve orientation invariant representation since the magnitude part is independent of the robot's direction. Fourier components obtained for each image by the pre-processing step are used to learn SFA representations. The learning phase has two steps; the first reduces dimensionality using a linear SFA, while the second extract non-linear slow features using a quadratic SFA. The input and output dimensionality for the first step are 450 and 20, respectively. The input and output dimensionality for the second step are 20 and 8, respectively.

We can visualize the SFA representation by plotting the color-coded SFA outputs for the images of all positions captured during the training phase. The theory has shown that for localization in open space with a relatively high rotational speed, the first two SFA outputs $s_{1,2}$ are orthogonal and monotonic (Franzius et al., 2007), so in this case, there is one global minimum for navigation. The features depicted in the figure 9.1 encode spatial information with a gradient along the coordinate axes of the

two slowest features. In contrast to the optimal solutions in open space (Franzius et al., 2007), the representations are distorted around the obstacle. Due to the slowness objective of SFA, views of temporally nearby data during training are encoded with similar slow feature values, while temporally distant data is encoded with distinct values due to the unit variance constraint of SFA. This implicitly encodes the information about obstacles present in a training area as the slow feature values on either side of an obstacle will be highly different. Figure 9.1 is based on simulation data. Unfortunately, providing such plots for the actual robot scenario is challenging since obtaining dense ground-truth data is computationally expensive due to the structure-from-motion (SfM) step. On the other hand, the quality of odometry information obtained during robot recordings is insufficient to plot such maps.

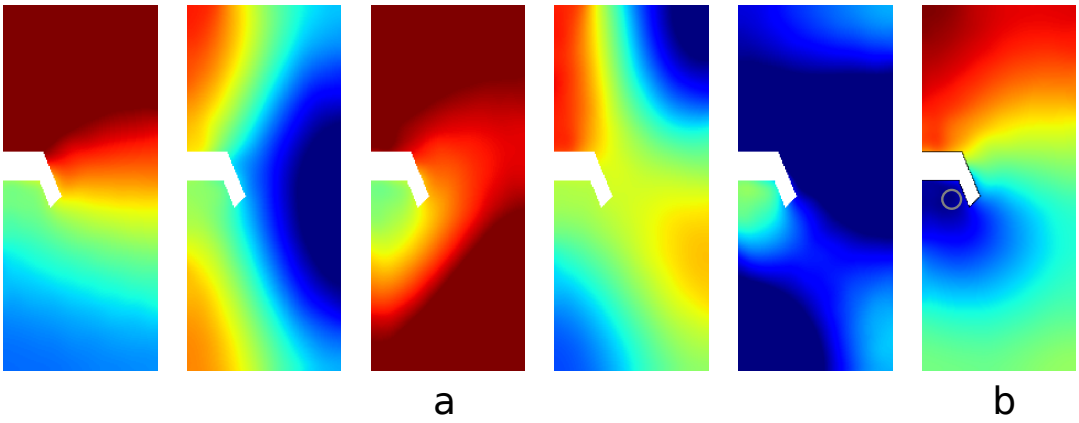


FIGURE 9.1: **Simulated Spatial Firing Maps:** (a) shows spatial firing maps of the first five SFA units $s_{1..5}$. The maps of the first two SFA $s_{1,2}$ units uniquely and smoothly encode the robot's position. The maps of functions $s_{3..5}$ show a mixture of the first two units and higher modes. (b) The plot shows an example cost surface in a slow feature space when the target position is in the convex region of the obstacle, as indicated by the circle. Performing gradient descent on this surface allows a robot to navigate around the obstacle.

9.3 Navigation Method

The system uses slow feature gradients to navigate between arbitrary points in a two-dimensional space. It is assumed that the target point's slow feature representation is known beforehand (e.g., stored during training). The *cost* function C is the Euclidean distance between slow feature representations of points in a 2D space. The system estimates the navigation direction by approximating the gradient of the cost surface. Thus, navigation between any two points can be achieved by performing gradient descent on the cost surface C. For an n-dimensional slow feature space, the mapping function $f : \mathbb{R}^2 \mapsto \mathbb{R}^n$ maps a position to the slow feature space by processing the associated image. The cost function ($C : \mathbb{R}^n \mapsto \mathbb{R}$), which computes the Euclidean

distance from the current position $p := (x_p, y_p)$ to the target position $t := (x_t, y_t)$ using only $f(p)$ as input (Metka et al., 2017), is given by:

$$C(f(p)) = \sqrt{\sum_{i=1}^{n} (f(p)_i - f(t)_i)^2}$$

However, the system computes a local linear approximation of the gradient as the analytical gradient $\frac{\partial C(f(p))}{\partial p}$ is infeasible to obtain. Here, the system uses robotic odometry to estimate the metric position of previous observations. Please note that odometry for the recent past is quite precise but deteriorates quickly over long distances.

9.3.1 Implementation

In general, to estimate the gradient direction at any given position p, in addition to the cost measurement at current position $C(f(p))$, we at least need the cost measurements from two nearby positions $C(f(p_1))$ and $C(f(p_2))$. These points must be non-collinear and are used to fit a plane to the surface of C. In contrast to previous work (Metka et al., 2017), where two cameras were used to obtain two measurements at each time step, only a single omnidirectional camera is mounted on a mobile robot in the current setting. Therefore, obtaining a gradient estimation directly from the starting point of navigation is impossible. For this reason, the robot traverses a fixed V-shaped initial trajectory and captures images during traversal. The next step is to compute the cost at each position by transforming all the captured images in the slow feature space. The following step creates an estimation matrix E that contains the coordinates (x, y) obtained from the robot's odometry information and the associated cost value C for each visited position. This is followed by applying Singular Value Decomposition (SVD) on the estimation matrix E to compute the gradient direction. Please note that although three non-collinear points are sufficient to estimate a gradient at any given position p, the system uses all the intermediate points to increase the robustness of gradient estimation. The constructed estimation matrix E for each traversed segment on the navigation path is stored as a history of past values. This information is also considered to estimate future gradients depending on the window size. The window size depends on the experiment's scenario (i.e., navigation in an open field or around an obstacle). The system parametrizes the window size by keeping the entire history for free area navigation and a subset for obstacle circumnavigation. Using a smaller window size for the latter case allows the robot to take sharp turns; otherwise, it would result in averaging the gradients.

The odometry quality degrades over time. However, it is still suitable for local position estimation, which the robot uses for motor control. After gradient estimation, the robot moves along the estimated gradient and captures all the intermediate images. A cost value threshold in slow feature space and a maximum number of iterations serve as stopping criteria. The process is repeated until one of the conditions is met. The estimated gradient is normalized and multiplied with a scaling factor η, and a momentum term γ is also used to incorporate information from past gradients to improve convergence (Metka et al., 2017). The navigation algorithm presented in the text uses a previously learned slow feature representation and a known target representation (see algorithm 1).

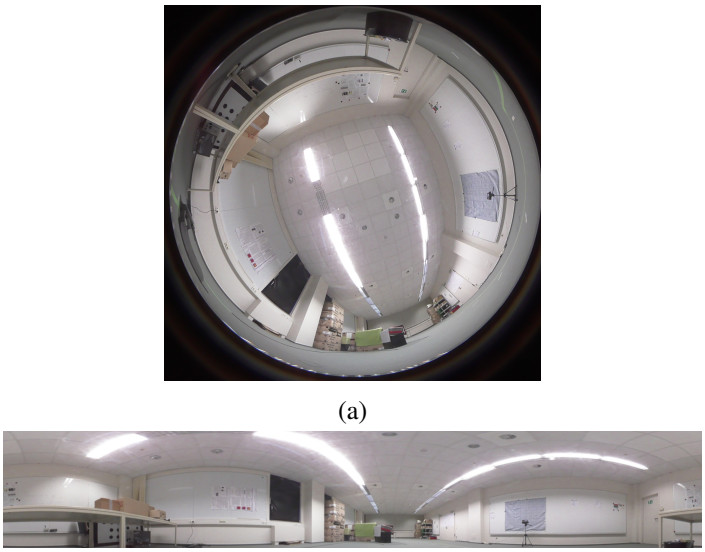
1 Navigation Algorithm

$\gamma = 0.3$ ▷ momentum term
$\eta = 0.5$ ▷ gradient scaling factor
previous_gradient = None
Traverse a fixed V-shaped trajectory of length 1.96 m and capture images I
do
    for each image *i* in I **do**
        k = project omnidirectional image *i* to a panoramic image
        p = row-wise Fourier expansion of *k*
        c = compute cost $C(f(p))$
        x,y = retrieve camera position from odometry information
        append (x,y,c) to estimation matrix E
    end for
    g = apply SVD on E to compute direction of steepest descent
    normalized_gradient = normalize g
    if previous_gradient is not None **then**
        gradient = γ * previous_gradient - η * normalized_gradient
    else
        gradient = η * normalized_gradient
    end if
    previous_gradient = gradient
    Move along the gradient direction and capture images I
    c = compute cost $C(f(p))$ at the updated position
while c $<$ 0.3 or number_of_iterations == 200

9.4 Experiments

Experiments were performed indoors with an area of 4×10 meters. A V-shaped obstacle was placed inside the training area to test the navigation performance around obstacles. Like the previous experiments, the modified lawn mower equipped with a single omnidirectional camera was used. It stores omnidirectional images of size

2880×2880 pixels. The system later projects the recorded images to corresponding panoramic views of size 300×30 pixels. Figure 9.2 shows an omnidirectional image and its associated panoramic image from the indoor environment. The lawn mower was put in a free mowing mode for the training phase. It traversed the environment for approximately 50 minutes to ensure an even sampling of the environment and captured $15,000$ images. In this mode, it drives straight segments until it detects the border, where it turns in a random direction. Figure 9.3 shows an example robot trajectory during the training run. Once the training phase is finished, slow features are computed instantaneously from Fourier components of a single image. The system only used the first two slow feature unit values $s_{1,2}$ to perform navigation. For all the experiments, the gradient scaling factor and the momentum are set to $\eta = 0.5$ and $\gamma = 0.3$. The maximum number of iterations and cost value threshold in slow feature space are set to 200 and 0.3 as stopping criteria, respectively.



(a)

(b)

FIGURE 9.2: **Real-world Indoor Environment**: (a) Example omnidirectional image from the indoor scene. (b) The corresponding panoramic image.

9.4.1 Navigation in a Free Space

This experiment aims to test the navigation in open spaces. In this case, the shortest path between the start and end positions does not contain any obstacle. The gradient-based navigation results are presented for four different start and end positions with five trials each.

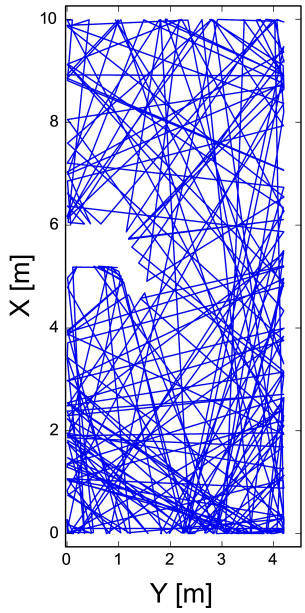


FIGURE 9.3: An example training trajectory consists of straight line segments with random orientation. A V-shaped obstacle is present in the field.

Results

Figure 9.4 shows the results of four different start and end positions. In all the experiments, the robot always reached within the close vicinity of the target location. However, it stopped at different distances from the target location, even for the trials of identical start and end positions. Thus, the goal approach rate serves as a performance measure. We varied the success criterion, the final distance to the target location between 0 and 1 meter, and plotted the goal approach rate for all test runs (fig. 9.5). The goal approach rate varies between 0% to 96%. For a final distance to the target location of 0.8 meters as successful navigation, the goal approach rate is 85%.

9.4.2 Navigation around an Obstacle

The aim here is to validate that the slow feature gradients allow a robot to navigate around obstacles without the need for explicit path planning. In these experiments, the target location was kept fixed inside the convex region of the obstacle while different start positions were chosen from the opposite side of the obstacle. The experimental results are presented for four different starting positions with five trials each.

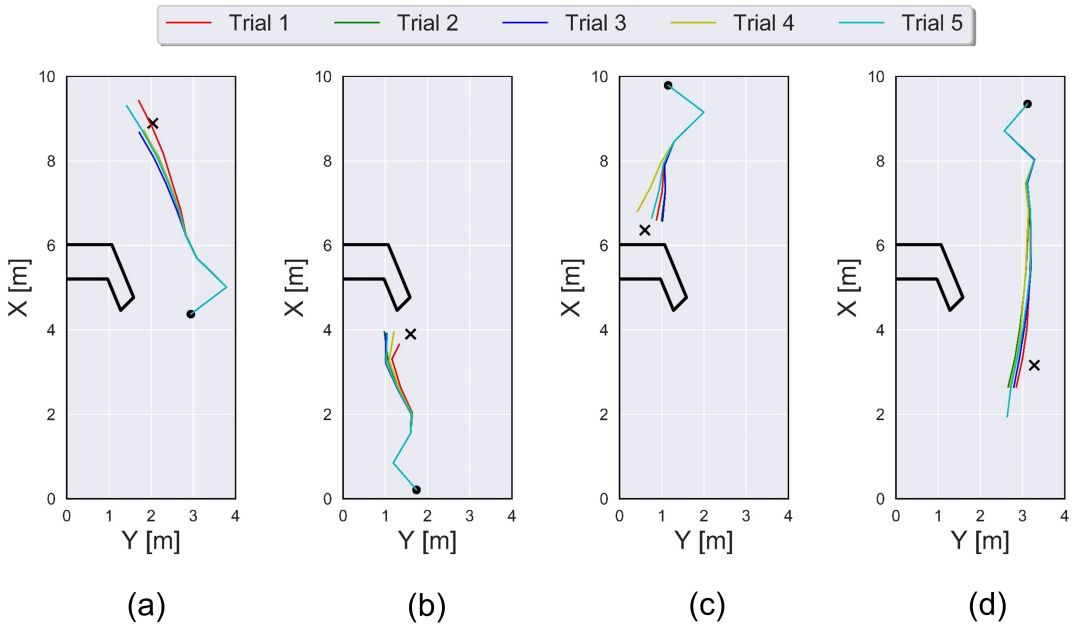


FIGURE 9.4: **Navigation in a Free Space**: A dot and cross, respectively, mark the start and target positions. (a)-(d) Results of four different start and end positions using the first two slow feature units $s_{1,2}$. A successful trial has a final distance of fewer than 0.8 meters from a target location.

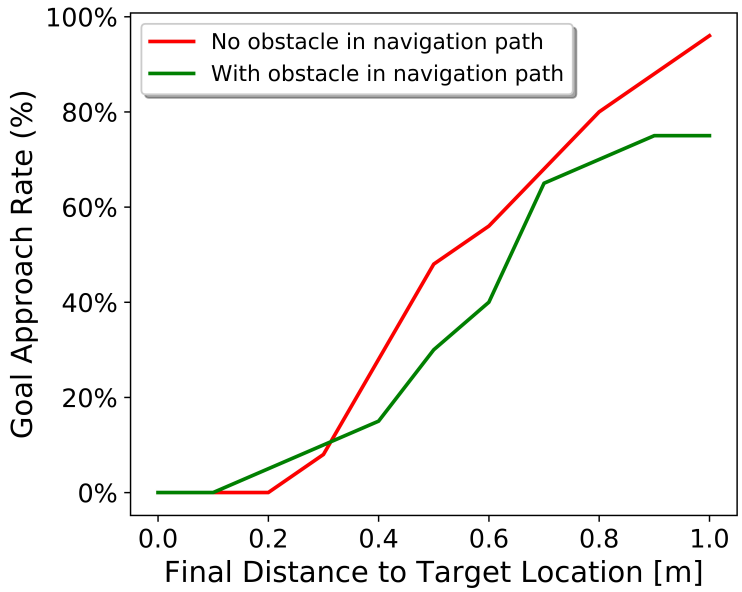


FIGURE 9.5: **Goal Approach Rate**: The red line shows the performance trade-off for free space navigation, while the green line shows the performance trade-off for obstacle circumnavigation. For the criterion of 0.8 meters as the final distance to the target location, the robot reached the target location in 85% of the trials for free space and 70% of the trials for circumnavigation.

Results

The resulting 20 trajectories are shown in fig. 9.6. The system repeated the same analysis performed for free area navigation experiments. The goal approach rate

varies between 0% to 75%. For a success criterion of 0.8 meters, the goal approach rate is 70%. The plot for the goal approach rate is shown in fig. 9.5 (green). The performance decrease comes from the cases of the robot navigating very closely and colliding with the obstacle. The robot hit the obstacle in these cases and never navigated to the target location. We encountered four such cases out of a total of twenty test runs. For the current analysis, we treated those cases as a failure. Another reason for failed navigation is that the robot got stuck in local minima due to flat gradients as the most variance is concentrated in the regions near the obstacle (see fig. 9.1b).

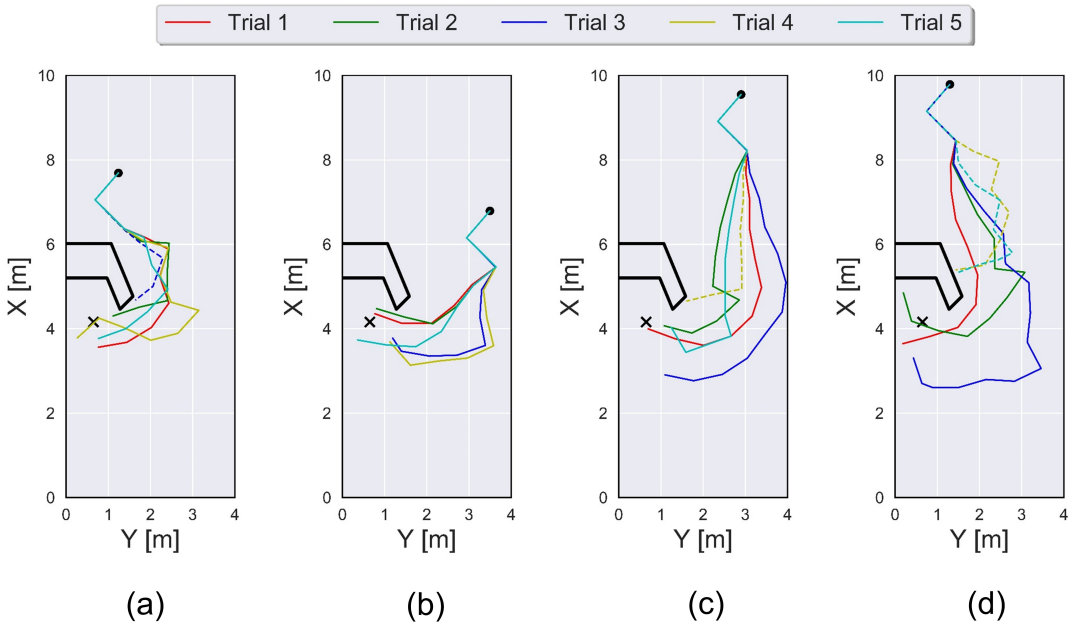


FIGURE 9.6: **Navigation around an Obstacle**: The start and target positions are marked by a dot and a cross, respectively. (a)-(d) Four different start positions with five trials each. The robot navigated the obstacle using the first two slow feature unit values $s_{1,2}$. The trajectories for the cases where the robot hit the obstacle are indicated by dashed lines. A successful trial has a final distance of fewer than 0.8 meters from a target location.

9.5 Conclusion

This chapter presented a robot navigation system that works directly in slow feature space using gradient descent. The slow feature representations are learned for the specific environment during an offline learning phase where the robot follows the standard movement pattern of a robotic lawn mower. After the unsupervised learning step, the robot can navigate by following the difference gradient between its current spatial representation and the target representation. The direction of the steepest descent is estimated from images in the recently traveled past. Locations not separated by obstacles were typically visited temporally close during training and thus

encoded with similar Slow Feature values. On the other hand, views from different sides of an obstacle were never seen in temporal proximity and got more different representations. Thus, the resulting slow feature representations implicitly encode average travel time during exploration (e.g., obstacles). Hence, circumnavigating obstacles requires no explicit path planning but is accomplished simply by following the steepest gradient. The robot reached the target in 85% of trials in an open field scenario and 70% when the target location was behind the obstacle. Hence, the proposed method reproduces the navigation results of extensive statistical experiments from a noise-free simulator environment (Metka et al., 2017). The variability in each set of trajectories is probably due to slight variances in start pose and environmental changes. In some failure cases, the robot got stuck in regions with flat gradients. Due to the presence of the obstacle in the training area, the resulting representations distort around it, which leads to relatively flat gradients for large parts of the training area, as also depicted in the simulated experiments (Metka et al., 2017). A more advanced gradient descent algorithm could help to cope with such issues. Further, an extension to the SFA algorithm made in (Richthofer et al., 2018) presents a feasible solution to overcome this issue. Other failures occurred when the robot hit the obstacle. In the future, the robot may fall back to standard behavior in such a case, drive a short distance in a random direction and then resume navigation. Alternatively, an obstacle avoidance sensor on the robot may be used to keep a minimum distance from obstacles. Environmental changes may directly influence spatial representations. However, we can improve them by using the strategies presented in the chapter 6.

The computational load for the proposed system is orders of magnitudes lower than that of standard SLAM or Deep Learning systems, as well as those based on Slow Feature Hierarchies. Learning and application phases are suitable for small and cheap systems based on digital signal processors (DSPs). The method is especially suited for an application where the robot has a default operation mode with random navigation during which the SFA learning can occur. Over time, the robot's efficiency and capabilities can improve as described.

Chapter 10

Summary and Conclusion

This thesis addressed the most fundamental and challenging problem of visual localization in unstructured outdoor environments. Visual localization estimates the position of an entity in an environment using a camera. Localization is an essential prerequisite for autonomous mobile robots, self-driving cars, and augmented reality applications. In static indoor environments, localization can be considered a solved problem. However, outdoor scenarios where a scene rapidly changes due to several factors (e.g., lighting, weather, seasons, and dynamic objects) pose a significant challenge to solutions tailored towards visual localization. Hence, this issue limits the application of most visual localization approaches in real-world scenarios. Another important aspect is the hardware requirement, as most state-of-the-art methods require a GPU to generate a scene representation and perform localization. This constraint may also limit the applicability of these methods on robots equipped with low-cost embedded hardware, for instance, lawnmowers. This thesis aimed to tackle both challenges by learning a spatial representation that allows an agent to localize itself over extended periods and using computationally inexpensive algorithms that do not require specialized hardware (e.g., GPU).

This work employed a bio-inspired model for visual localization as many animals (e.g., rats) show excellent localization and navigation abilities in natural environments. The model can reproduce the firing characteristics of Place and Head-Direction Cells found in the rat brain when trained with an agent's visual cues. The model uses the concept of unsupervised Slow Features Analysis (SFA), which states that behaviorally meaningful information (e.g., position or orientation of an animal in space) changes on a slower timescale compared to the primary sensory input (e.g., pixel values in a video). The encoding of a particular cell type in the learned representation depends on the agent's movement statistics during training. If, for instance, its position changes on a slower time scale than other variables (e.g., orientation), then the learned representation will implicitly code for the agent's position in space. Thus the model allows learning relevant representation (here: robot's position) directly from

input statistics in an unsupervised learning process.
The proposed mapping pipeline uses an image stream collected during a robot recording session to learn the spatial representation of a scene. The pipeline applies SFA to the training data and extracts slow features. For localization, these features must ideally encode the robot's position (x,y) in space. After training, the localization step is instantaneous, i.e., the model only needs a single image to compute the output. The localization pipeline uses the trained SFA model to compute output in the SFA space. This work shows that SFA space is sufficient for navigation without explicit path planning. However, a supervised post-processing step maps SFA to metric space for evaluating and comparing the proposed methods to other localization approaches. After this step, the pipeline outputs the 2D position (x,y) of each test image.
In outdoor scenarios changing conditions have a substantial impact on the appearance of a scene, which often prevents successful visual localization. The application of SFA on the images captured by a robot enables self-localization from a single image. However, changes occurring during the training phase or over an extended period can affect the learned representation. This work proposed to join long-term robot recordings based on their position correspondences to address the problem. The restructuring scheme allows generating a training sequence where environmental conditions vary faster than the position of a robot. The generated training data enables SFA to learn invariance to changing conditions. The experiments were performed using data from a simulated and real-world outdoor environment. The outdoor data was collected over an entire year with effects like different daytime, weather, seasons, and dynamic objects. Results show an increasing invariance w.r.t changing conditions over time. Thus an outdoor robot can improve its localization performance during operation. The established hierarchical SFA model trained on raw images performs well. However, using Fourier features to learn a scene representation reduces the computation time and makes it adequate to run on an ARM embedded system.
Instead of using complete image information, an alternative is to perform visual localization relative to landmarks present in an environment. The proposed system learns to recognize landmarks in images and then uses SFA for unsupervised position estimation w.r.t to each landmark. The straightforward way to learn new landmarks in a scene is using hand-labeled data to train a neural network. However, this process is tedious and costly. The proposed approach allows a robot to learn landmarks for localization with a human cooperatively. This approach uses pre-trained detectors of everyday objects to learn new landmarks in a scene, requiring minimal human supervision. Hence, the method bootstraps the landmark learning process and removes the need to label large amounts of data manually. The human teacher has complete control over selecting new landmarks, allowing learning of unique, robustly detectable,

and semantic landmarks. The work presented localization results using the learned landmarks in simulated and real-world outdoor environments and compared the results to models based on complete images and PoseNet. The landmark-based localization achieved better accuracy than the baseline methods in a challenging large-scale environment. Moreover, the results show that localization accuracy increases with the number of learned landmarks. Hence the landmark-based approach enables large-scale localization, an essential feature of any localization method. However, the only limitation is its dependency on the deep learning-based pre-processing step, which requires a GPU for execution. Nevertheless, the introduced approach for learning generic landmarks can overcome this limitation. The proposed method allows encoding of the robot's gaze as slowly varying features by utilizing the invariance learning capability of SFA. After the training phase, the trained model can be used to perform localization relative to any image patch. This work presented the results of its prototype implementation, which seems to be quite promising and on par with other SFA-based approaches. Future work will investigate the integration of a stable visual tracker for generating the desired training sequence, allowing the robot to perform localization by looking anywhere in the scene.

State-of-the-art methods use the 3D structure of a scene for precise visual localization. However, 3D scene reconstruction is resource-intensive in terms of hardware requirements and computation time, making it infeasible to run on low-cost embedded hardware. Unsupervised spatial representation learning with SFA enables computationally inexpensive localization and mapping. This work compared the SFA-based approach with the well-known structure-based Active Search and learning-based PoseNet methods in two distinct settings: short-term temporal and extreme spatial generalization. Results show that the SFA-based mapping and localization are drastically faster than Active Search and PoseNet while achieving comparable localization accuracy in the test scenario.

As mentioned earlier, the learned SFA representation is sufficient for the navigation task. After the unsupervised learning phase, a subset of the resulting representation encodes the robot's position. The representation is spatially smooth and implicitly encodes the average travel time during exploration. Following the SFA gradient allows the robot to navigate even around obstacles without any planning. Earlier work showed this basic principle in noise-free simulation, using two virtual cameras on a robot. This work extended the approach to be more robust and computationally efficient. The experiments were performed on a lawn mower robot with a single camera in an indoor environment. The results show successful navigation in open spaces and around obstacles using slow feature gradients. The following section summarizes the main takeaways of the proposed methods in the context of this thesis.

10.1 Scientific Impact

The unsupervised learning for visual localization and navigation offers the following advantages over the state-of-the-art approaches:

- **Computational Efficiency**: Most state-of-the-art approaches (both structure- and learning-based) require high-end GPUs to generate the environment's map. Thus these methods are not best-suited for low-powered embedded devices. The SFA-based approach does not require high-end hardware for computation and is up to 800 times faster than the state-of-the-art structure-based method (Haris et al., 2021).
- **Sensor Calibration**: State-of-the-art methods that rely on the underlying geometry of a scene require an offline calibration phase to estimate the camera's parameters. Generally, it is assumed that the intrinsic parameters will remain fixed over time. However, several environmental factors (e.g., temperature and humidity) may invalidate the previously obtained calibration. On the other hand, learning-based approaches operate without sensor calibration.
- **Fault Detection**: Due to the slowness objective of SFA, temporally close/sparse data gets encoded by similar/unique values, which leads to a spatially smooth representation. This property makes it easy to detect values that are highly different than temporally close data. Thus the approach provides a simple way to detect localization errors.
- **Robustness to Environmental Changes**: Most SLAM systems operate assuming that the world will remain unchanged during the entire robot operation (Cadena et al., 2016). While this may hold for small-scale indoor environments, it is impossible for larger or dynamic scenes. These methods require map building from scratch or updating certain map parts with human intervention in changing conditions. On the other hand, SFA can learn invariance to such changes by restructuring the training data, as introduced in (Haris et al., 2019), without any human intervention.
- **Scale Invariance**: Traditional SLAM systems that use a single camera suffers from the scale ambiguity problem and estimate the structure and camera's pose with an undetermined scale factor. Scale is typically recovered as a post-processing step if the absolute distance between two points in space is known. The SFA approach works independently of the absolute scale of the environment.

- **High-level Behavior**: SFA-based approach has potential for higher-level intelligent behavior (e.g., navigation and efficient planning-free obstacle avoidance in slow feature space (Haris et al., 2018)).
- **Labeled Data**: Supervised end-to-end learning approaches need labeled data for the training phase, typically obtained with a computationally expensive structure-from-motion (SfM) step. On the other hand, the method here learns a spatial representation of the environment without labeled data.
- **Minimal Assumptions on the Environment**: Most geometric-based methods struggle when the environment has too few features. As long as the environment is suitably rich (i.e., without an identical appearance at different locations), SFA can encode a unique representation for each position in space.

However, there are limitations and disadvantages:

- **Offline Method**: SLAM systems build the environment map and perform the localization concurrently; SFA requires an offline mapping phase before localization.
- **Camera**: So far, the best results are achieved with an up-facing fisheye camera or omnidirectional camera, but the technique can be applied to perspective cameras.

All localization and navigation experiments were recorded on an outdoor mobile robot and demonstrated in actual working conditions for autonomous lawnmowers and indoor service robots. Robustness to changing outdoor conditions is a significant part of the approach, and the computational efficiency of the method makes it unique for implementation on potential low-cost products. To conclude, the proposed technique offers robust localization and navigation with a straightforward model, and it is feasible for service robots (i.e., lawnmowers and vacuum cleaners) typically equipped with low-cost embedded hardware.

Bibliography

Arandjelovic, Relja and Andrew Zisserman (2014). "DisLocation: Scalable Descriptor Distinctiveness for Location Recognition". In: *ACCV*.

Arleo, Angelo and Wulfram Gerstner (2000). "Spatial cognition and neuro-mimetic navigation: a model of hippocampal place cell activity". In: *Biological Cybernetics* 83.3, pp. 287–299. DOI: 10.1007/s004220000171.

Bailey, T. and H. Durrant-Whyte (2006). "Simultaneous localization and mapping (SLAM): part II". In: *IEEE Robotics and Automation Magazine* 13.3, pp. 108–117. DOI: 10.1109/MRA.2006.1678144.

Balntas, Vassileios, Shuda Li, and Victor Prisacariu (2018). "RelocNet: Continuous Metric Learning Relocalisation using Neural Nets". In: *The European Conference on Computer Vision (ECCV)*.

Bao, S. Y., M. Bagra, Y. Chao, and S. Savarese (2012). "Semantic structure from motion with points, regions, and objects". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2703–2710.

Barraquand, J. and J. Latombe (1991). "Robot Motion Planning: A Distributed Representation Approach". In: *I. J. Robotics Res.* 10.6, pp. 628–649.

Barrera, Alejandra and Alfredo Weitzenfeld (2008). "Biologically-inspired robot spatial cognition based on rat neurophysiological studies". In: *Autonomous Robots* 25.1, pp. 147–169. DOI: 10.1007/s10514-007-9074-3.

Bay, Herbert, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool (2008). "Speeded-Up Robust Features (SURF)". In: *Computer Vision and Image Understanding*, pp. 346–359. DOI: https://doi.org/10.1016/j.cviu.2007.09.014.

Böhmer, Wendelin, Steffen Grünewälder, Yun Shen, Marek Musial, and Klaus Obermayer (2013). "Construction of approximation spaces for reinforcement learning". In: *Journal of Machine Learning Research* 14.1, pp. 2067–2118.

Bowman, S. L., N. Atanasov, K. Daniilidis, and G. J. Pappas (2017). "Probabilistic data association for semantic SLAM". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1722–1729.

Brachmann, Eric, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother (2017). "Dsac-differentiable ransac for camera localization". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6684–6692.

Cadena, Cesar, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J. Leonard (2016). "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age". In: *IEEE Transactions on Robotics* 32.6, pp. 1309–1332. DOI: 10.1109/TRO.2016.2624754.

Campos, Carlos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós (2021). "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM". In: *IEEE Transactions on Robotics* 37.6, pp. 1874–1890. DOI: 10.1109/TRO.2021.3075644.

Carlevaris-Bianco, Nicholas and Ryan M. Eustice (2014). "Learning visual feature descriptors for dynamic lighting conditions". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2769–2776. DOI: 10.1109/IROS.2014.6942941.

Chen, Shitao, Songyi Zhang, Jinghao Shang, Badong Chen, and Nanning Zheng (2019). "Brain-Inspired Cognitive Model With Attention for Self-Driving Cars". In: *IEEE Transactions on Cognitive and Developmental Systems* 11.1, pp. 13–25. DOI: 10.1109/TCDS.2017.2717451.

Chum, Ondřej and Jiri Matas (2008). "Optimal Randomized RANSAC". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, pp. 1472–1482.

Churchill, Winston and Paul Newman (2013). "Experience-based navigation for long-term localisation". In: *International Journal of Robotics Research* 32.14, pp. 1645–1661. DOI: 10.1177/0278364913499193.

Collett, Matthew, Lars Chittka, and Thomas S. Collett (2013). "Spatial Memory in Insect Navigation". In: *Current Biology* 23.17, R789 –R800. DOI: https://doi.org/10.1016/j.cub.2013.07.020.

Cuperlier, Nicolas, Mathias Quoy, and Philippe Gaussier (2007). "Neurobiologically inspired mobile robot navigation and planning". In: *Frontiers in Neurorobotics* 1. DOI: 10.3389/neuro.12.003.2007.

Davison, Andrew J., Ian D. Reid, Nicholas D. Molton, and Olivier Stasse (2007). "MonoSLAM: Real-Time Single Camera SLAM". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.6, pp. 1052–1067. DOI: 10.1109/TPAMI.2007.1049.

DeTone, Daniel, Tomasz Malisiewicz, and Andrew Rabinovich (2017). "SuperPoint: Self-Supervised Interest Point Detection and Description". In: *CoRR*. arXiv: 1712.07629.

Durrant-Whyte, H. and T. Bailey (2006). "Simultaneous localization and mapping: part I". In: *IEEE Robotics and Automation Magazine* 13.2, pp. 99–110. DOI: 10.1109/MRA.2006.1638022.

Dusmanu, Mihai, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler (2019). "D2-Net: A Trainable CNN for Joint Detection and Description of Local Features". In: *CVPR 2019 - IEEE Conference on Computer Vision and Pattern Recognition*. URL: https://hal.archives-ouvertes.fr/hal-02438461.

Einecke, Nils, Jörg Deigmöller, Keiji Muro, and Mathias Franzius (2018). "Boundary Wire Mapping on Autonomous Lawn Mowers". In: *Field and Service Robotics*. Springer International Publishing, pp. 351–365. ISBN: 978-3-319-67361-5.

Engel, Jakob, Vladlen Koltun, and Daniel Cremers (2018). "Direct Sparse Odometry". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.3, pp. 611–625. DOI: 10.1109/TPAMI.2017.2658577.

Engel, Jakob J., Thomas Schöps, and Daniel Cremers (2014). "LSD-SLAM: Large-Scale Direct Monocular SLAM". In: *ECCV*.

Espada, Yoan, Nicolas Cuperlier, Guillaume Bresson, and Olivier Romain (2019). "From Neurorobotic Localization to Autonomous Vehicles". In: *Unmanned systems* 07.03, pp. 183–194. DOI: 10.1142/S2301385019410048.

Forster, Christian, Matia Pizzoli, and Davide Scaramuzza (2014). "SVO: Fast semi-direct monocular visual odometry". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 15–22. DOI: 10.1109/ICRA.2014.6906584.

Franzius, Mathias, Henning Sprekeler, and Laurenz Wiskott (2007). "Slowness and Sparseness Lead to Place, Head-Direction, and Spatial-View Cells". In: *PLOS Computational Biology*, pp. 1–18. DOI: 10.1371/journal.pcbi.0030166.

Frost, D., V. Prisacariu, and D. Murray (2018). "Recovering Stable Scale in Monocular SLAM Using Object-Supplemented Bundle Adjustment". In: *IEEE Transactions on Robotics* 34.3, pp. 736–747.

Fuentes-Pacheco, Jorge, José Ruiz-Ascencio, and Juan Manuel Rendón-Mancha (2015). "Visual simultaneous localization and mapping: a survey". In: *Artificial Intelligence Review* 43.1, pp. 55–81.

Galvez-López, Dorian and Juan D. Tardos (Oct. 2012). "Bags of Binary Words for Fast Place Recognition in Image Sequences". In: *Trans. Rob.* 28.5, 1188–1197. DOI: 10.1109/TRO.2012.2197158.

Gálvez-López, Dorian, Marta Salas, Juan D. Tardós, and J.M.M. Montiel (2016). "Real-Time Monocular Object SLAM". In: *Robot. Auton. Syst.* 75, 435–449. DOI: 10.1016/j.robot.2015.08.009.

Galvez-López, Dorian and Juan D. Tardos (2012). "Bags of Binary Words for Fast Place Recognition in Image Sequences". In: *IEEE Transactions on Robotics* 28.5, pp. 1188–1197. DOI: 10.1109/TRO.2012.2197158.

Garrido-Jurado, S., R. Muñoz-Salinas, F.J. Madrid-Cuevas, and M.J. Marín-Jiménez (2014). "Automatic generation and detection of highly reliable fiducial markers under occlusion". In: *Pattern Recognition* 47.6, pp. 2280–2292. DOI: https://doi.org/10.1016/j.patcog.2014.01.005.

Gioi, Rafael Grompone von, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall (2010). "LSD: A Fast Line Segment Detector with a False Detection Control". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.4, pp. 722–732. DOI: 10.1109/TPAMI.2008.300.

Gomez-Ojeda, Ruben, Francisco-Angel Moreno, David Zuñiga-Noël, Davide Scaramuzza, and Javier Gonzalez-Jimenez (2019). "PL-SLAM: A Stereo SLAM System Through the Combination of Points and Line Segments". In: *IEEE Transactions on Robotics* 35.3, pp. 734–746. DOI: 10.1109/TRO.2019.2899783.

Haris, Muhammad, Mathias Franzius, and Ute Bauer-Wersing (2018). "Robot Navigation on Slow Feature Gradients". In: *Neural Information Processing*. Springer International Publishing, pp. 143–154. ISBN: 978-3-030-04239-4.

– (2019). "Robust Outdoor Self-localization In Changing Environments". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 714–719. DOI: 10.1109/IROS40897.2019.8967549.

– (2021). "Unsupervised Fast Visual Localization and Mapping with Slow Features". In: *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 519–523. DOI: 10.1109/ICIP42928.2021.9506656.

Hart, Peter E., Nils J. Nilsson, and Bertram Raphael (1968). "A formal basis for the heuristic determination of minimum cost paths". In: *IEEE Transactions on Systems Science and Cybernetics* SSC-4(2), pp. 100–107.

Hosseinzadeh, M., K. Li, Y. Latif, and I. Reid (2019). "Real-Time Monocular Object-Model Aware Sparse SLAM". In: *2019 International Conference on Robotics and Automation (ICRA)*, pp. 7123–7129.

Humenberger, Martin, Yohann Cabon, Nicolas Guérin, Julien Morat, Jérôme Revaud, Philippe Rerole, Noé Pion, César Roberto de Souza, Vincent Leroy, and Gabriela Csurka (2020). "Robust Image Retrieval-based Visual Localization using Kapture". In: *CoRR*. URL: https://arxiv.org/abs/2007.13867.

Ishiguro, H. and S. Tsuji (1996). "Image-based memory of environment". In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS '96*. Vol. 2, 634–639 vol.2. DOI: 10.1109/IROS.1996.571018.

Jeffery, Kathryn J. and John M. O'Keefe (1999). "Learned interaction of visual and idiothetic cues in the control of place field orientation". In: *Experimental Brain Research* 127, pp. 151–161.

Kendall, Alex, Matthew Grimes, and Roberto Cipolla (2015). "Convolutional networks for real-time 6-DOF camera relocalization". In: *CoRR*. URL: http://arxiv.org/abs/1505.07427.

Kendall, Alex and Roberto Cipolla (2017). "Geometric Loss Functions for Camera Pose Regression with Deep Learning". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6555–6564. DOI: 10.1109/CVPR.2017.694.

Khatib, O. (1985). "Real-time obstacle avoidance for manipulators and mobile robots". In: *Proceedings. 1985 IEEE International Conference on Robotics and Automation*. Vol. 2, pp. 500–505.

Klein, Georg and David Murray (2007). "Parallel Tracking and Mapping for Small AR Workspaces". In: *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pp. 225–234. DOI: 10.1109/ISMAR.2007.4538852.

Kneip, Laurent, Davide Scaramuzza, and Roland Siegwart (2011). "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation". In: *CVPR 2011*, pp. 2969–2976. DOI: 10.1109/CVPR.2011.5995464.

Knierim, JJ, HS Kudrimoti, and BL McNaughton (1995). "Place cells, head direction cells, and the learning of landmark stability". In: *Journal of Neuroscience* 15.3, pp. 1648–1659. DOI: 10.1523/JNEUROSCI.15-03-01648.1995.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey Hinton (Jan. 2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Neural Information Processing Systems* 25. DOI: 10.1145/3065386.

Lambrinos, Dimitrios, Ralf Möller, Thomas Labhart, Rolf Pfeifer, and Rüdiger Wehner (2000). "A mobile robot employing insect strategies for navigation". In: *Robotics and Autonomous Systems* 30.1-2, pp. 39–64. DOI: 10.1016/S0921-8890(99)00064-0.

Legenstein, Robert, Niko Wilbert, and Laurenz Wiskott (2010). "Reinforcement Learning on Slow Features of High-Dimensional Input Streams". In: *PLoS Computational Biology* 6.8, pp. 1–13.

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick (2014). "Microsoft COCO: Common Objects in Context". In: *Computer Vision – ECCV 2014*. Springer International Publishing, pp. 740–755.

Liu, Liu, Hongdong Li, and Yuchao Dai (2017). "Efficient Global 2D-3D Matching for Camera Localization in a Large-Scale 3D Map". In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2391–2400. DOI: 10.1109/ICCV.2017.260.

Lowe, David G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints". In: *International Journal of Computer Vision* 60, pp. 91–110.

Mccormac, John, Ronald Clark, Michael Bloesch, Andrew Davison, and Stefan Leutenegger (2018). "Fusion++: Volumetric Object-Level SLAM". In: pp. 32–41. DOI: 10.1109/3DV.2018.00015.

McManus, Colin, Winston Churchill, Will Maddern, Alexander D. Stewart, and Paul Newman (2014). "Shady dealings: Robust, long-term visual localisation using illumination invariance". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 901–906. DOI: 10.1109/ICRA.2014.6906961.

Menegatti, Emanuele, Mauro Zoccarato, Enrico Pagello, and Hiroshi Ishiguro (2003). "Image-Based Monte-Carlo Localisation without a Map". In: *AI*IA 2003: Advances in Artificial Intelligence*. Springer Berlin Heidelberg, pp. 423–435. ISBN: 978-3-540-39853-0.

Meng, L., J. Chen, F. Tung, J. J. Little, J. Valentin, and C. W. de Silva (2017). "Backtracking regression forests for accurate camera relocalization". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6886–6893. DOI: 10.1109/IROS.2017.8206611.

Metka, Benjamin (2019). "Robust Visual Self-localization and Navigation in Outdoor Environments Using Slow Feature Analysis". In:

Metka, Benjamin, Mathias Franzius, and Ute Bauer-Wersing (2013). "Outdoor Self-Localization of a Mobile Robot Using Slow Feature Analysis". In: *Neural Information Processing - 20th International Conference, ICONIP 2013, Daegu, Korea, November 3-7, 2013. Proceedings, Part I*, pp. 249–256.

– (2016). "Improving Robustness of Slow Feature Analysis Based Localization Using Loop Closure Events". In: *Artificial Neural Networks and Machine Learning - ICANN 2016 - 25th International Conference on Artificial Neural Networks, Barcelona, Spain, September 6-9, 2016, Proceedings, Part II*. Vol. 9887. Lecture Notes in Computer Science. Springer, pp. 489–496. DOI: 10.1007/978-3-319-44781-0_58.

– (2017). “Efficient navigation using slow feature gradients”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*, pp. 1311–1316.

Metka, Benjamin, Mathias Franzius, and Ute Bauer-Wersing (Sept. 2018). “Bio-inspired visual self-localization in real world scenarios using Slow Feature Analysis”. In: *PLOS ONE* 13.9, pp. 1–18. DOI: 10.1371/journal.pone.0203994.

Meyer, Jean-Arcady and David Filliat (2003). “Map-based navigation in mobile robots. II. A review of map-learning and path-planning strategies”. In: *Cognitive Systems Research* 4.4, pp. 283–317.

Milford, Michael, Gordon Wyeth, and David Prasser (2004). “RatSLAM: a Hippocampal Model for Simultaneous Localization and Mapping”. In: *Proceedings of the 2004 IEEE International Conference on Robotics and Automation, ICRA 2004, April 26 - May 1, 2004, New Orleans, LA, USA*, pp. 403–408.

Milford, Michael and Gordon Wyeth (2010). “Persistent Navigation and Mapping using a Biologically Inspired SLAM System”. In: *I. J. Robotics Res.* 29.9, pp. 1131–1153.

Milford, Michael and Ruth Schulz (2014). “Principles of goal-directed spatial robot navigation in biomimetic models”. In: *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 369.1655. DOI: 10.1098/rstb.2013.0484.

Milford, Michael J. and Gordon F. Wyeth (2008). “Mapping a Suburb With a Single Camera Using a Biologically Inspired SLAM System”. In: *IEEE Transactions on Robotics* 24.5, pp. 1038–1053. DOI: 10.1109/TRO.2008.2004520.

Moulon, Pierre, Pascal Monasse, and Renaud Marlet (2013). “Global Fusion of Relative Motions for Robust, Accurate and Scalable Structure from Motion”. In: *2013 IEEE International Conference on Computer Vision*, pp. 3248–3255. DOI: 10.1109/ICCV.2013.403.

Muehlemann, Anton (2019). *TrainYourOwnYOLO: Building a Custom Object Detector from Scratch*. URL: https://github.com/AntonMu/.

Mueller, Matthias, Neil Smith, and Bernard Ghanem (2017). “Context-Aware Correlation Filter Tracking”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1387–1395. DOI: 10.1109/CVPR.2017.152.

Mur-Artal, Raúl, J. M. M. Montiel, and Juan D. Tardós (2015). “ORB-SLAM: A Versatile and Accurate Monocular SLAM System”. In: *IEEE Transactions on Robotics* 31.5, pp. 1147–1163. DOI: 10.1109/TRO.2015.2463671.

Mur-Artal, Raúl and Juan D. Tardós (2017). "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras". In: *IEEE Transactions on Robotics* 33.5, pp. 1255–1262. DOI: 10.1109/TRO.2017.2705103.

Newcombe, Richard A., S. Lovegrove, and Andrew J. Davison (2011). "DTAM: Dense tracking and mapping in real-time". In: *2011 International Conference on Computer Vision*, pp. 2320–2327.

Nicholson, Lachlan, Michael Milford, and Niko Sünderhauf (2018). "QuadricSLAM: Constrained Dual Quadrics from Object Detections as Landmarks in Semantic SLAM". In: *CoRR*. arXiv: 1804.04011.

O'Keefe, J. and J. Dostrovsky (1971). "The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat". In: *Brain Research* 34.1, pp. 171–175. DOI: https://doi.org/10.1016/0006-8993(71)90358-1.

Parkhiya, P., R. Khawad, J. K. Murthy, B. Bhowmick, and K. M. Krishna (2018). "Constructing Category-Specific Models for Monocular Object-SLAM". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4517–4524.

Philippides, Andrew, Bart Baddeley, Ken Cheng, and Paul Graham (2011). "How might ants use panoramic views for route navigation?" In: *Journal of Experimental Biology* 214.3, pp. 445–451. DOI: 10.1242/jeb.046755.

Pumarola, Albert, Alexander Vakhitov, Antonio Agudo, Alberto Sanfeliu, and Francese Moreno-Noguer (2017). "PL-SLAM: Real-time monocular visual SLAM with points and lines". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4503–4508. DOI: 10.1109/ICRA.2017.7989522.

Redmon, Joseph and Ali Farhadi (2018). "YOLOv3: An Incremental Improvement". In: *CoRR* abs/1804.02767. arXiv: 1804.02767.

Richthofer, Stefan and Laurenz Wiskott (2018). "Global Navigation Using Predictable and Slow Feature Analysis in Multiroom Environments, Path Planning and Other Control Tasks". In: *CoRR*.

Rublee, Ethan, Vincent Rabaud, Kurt Konolige, and Gary Bradski (2011). "ORB: An efficient alternative to SIFT or SURF". In: *2011 International Conference on Computer Vision*, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.

Salas-Moreno, Renato F., Richard A. Newcombe, Hauke Strasdat, Paul H.J. Kelly, and Andrew J. Davison (2013). "SLAM++: Simultaneous Localisation and Mapping at the Level of Objects". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sarlin, Paul-Edouard, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk (2019). "From Coarse to Fine: Robust Hierarchical Localization at Large Scale". In: *2019*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 12708–12717. DOI: 10.1109/CVPR.2019.01300.

Sattler, T., B. Leibe, and L. Kobbelt (2017). "Efficient Effective Prioritized Matching for Large-Scale Image-Based Localization". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.9, pp. 1744–1756. DOI: 10.1109/TPAMI.2016.2611662.

Sattler, Torsten, Bastian Leibe, and Leif Kobbelt (2011). "Fast image-based localization using direct 2D-to-3D matching". In: *2011 International Conference on Computer Vision*, pp. 667–674. DOI: 10.1109/ICCV.2011.6126302.

– (2012). "Improving Image-Based Localization by Active Correspondence Search". In: *Computer Vision – ECCV 2012*. Springer Berlin Heidelberg, pp. 752–765.

Sattler, Torsten, Qunjie Zhou, Marc Pollefeys, and Laura Leal-Taixé (2019). "Understanding the Limitations of CNN-Based Absolute Camera Pose Regression". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3297–3307.

Schönberger, Johannes L. and Jan-Michael Frahm (2016). "Structure-from-Motion Revisited". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4104–4113. DOI: 10.1109/CVPR.2016.445.

Schönberger, Johannes L., Hans Hardmeier, Torsten Sattler, and Marc Pollefeys (2017). "Comparative Evaluation of Hand-Crafted and Learned Local Features". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6959–6968. DOI: 10.1109/CVPR.2017.736.

Schönberger, Johannes L., Marc Pollefeys, Andreas Geiger, and Torsten Sattler (2018). "Semantic Visual Localization". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Shi, Jianbo and Tomasi (1994). "Good features to track". In: *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 593–600. DOI: 10.1109/CVPR.1994.323794.

Smith, Randall, Matthew Self, and Peter Cheeseman (1990). "Estimating Uncertain Spatial Relationships in Robotics". In: *Autonomous Robot Vehicles*. Springer New York, pp. 167–193. DOI: 10.1007/978-1-4613-8997-2_14.

Snavely, Noah, Steven M. Seitz, and Richard Szeliski (2006). "Photo tourism: Exploring photo collections in 3D". In: *SIGGRAPH Conference Proceedings*. ACM Press, pp. 835–846. ISBN: 1-59593-364-6.

Stone, Thomas, Michael Mangan, Paul Ardin, and Barbara Webb (2014). "Sky segmentation with ultraviolet images can be used for navigation". In: *Robotics: Science and Systems X, University of California, Berkeley, USA, July 12-16, 2014*. URL: http://www.roboticsproceedings.org/rss10/p47.html.

Sucar, E. and J. Hayet (2018). "Bayesian Scale Estimation for Monocular SLAM Based on Generic Object Detection for Correcting Scale Drift". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5152–5158.

Sünderhauf, Niko, Feras Dayoub, Sareh Shirazi, Ben Upcroft, and Michael Milford (2015). "On the Performance of ConvNet Features for Place Recognition". In: *CoRR*. URL: http://arxiv.org/abs/1501.04158.

Sünderhauf, Niko, Sareh Shirazi, Adam Jacobson, Feras Dayoub, Edward Pepperell, Ben Upcroft, and Michael Milford (2015). "Place recognition with ConvNet landmarks: Viewpoint-robust, condition-robust, training-free". In: *Robotics: Science and Systems*. URL: https://eprints.qut.edu.au/84931/.

Taube, Jeffrey, Robert Muller, and James Ranck Jr (1990). "Head-direction cells recorded from the postsubiculum in freely moving rats. I. Description and quantitative analysis". In: *The Journal of neuroscience* 10, pp. 420–35. DOI: 10.1523/JNEUROSCI.10-02-00420.1990.

Thrun, Sebastian (1998). "Bayesian Landmark Learning for Mobile Robot Localization". In: *Mach. Learn.* 33.1, 41–76. DOI: 10.1023/A:1007554531242.

Tilove, R. B. (1990). "Local obstacle avoidance for mobile robots based on the method of artificial potentials". In: *Proceedings., IEEE International Conference on Robotics and Automation*, 566–571 vol.1. DOI: 10.1109/ROBOT.1990.126041.

Toft, Carl et al. (2022). "Long-Term Visual Localization Revisited". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.4, pp. 2074–2088. DOI: 10.1109/TPAMI.2020.3032010.

Valada, Abhinav, Noha Radwan, and Wolfram Burgard (2018). "Deep Auxiliary Learning for Visual Localization and Odometry". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6939–6946. DOI: 10.1109/ICRA.2018.8462979.

Valgren, Christoffer and Achim J. Lilienthal (2010). "SIFT, SURF & seasons: Appearance-based long-term localization in outdoor environments". In: *Robotics and Autonomous Systems* 2, pp. 149–156. DOI: https://doi.org/10.1016/j.robot.2009.09.010.

Wehner, R, B Michel, and P Antonsen (1996). "Visual navigation in insects: coupling of egocentric and geocentric information". In: *Journal of Experimental Biology* 199.1, pp. 129–140. URL: http://jeb.biologists.org/content/199/1/129.

Wiskott, L., P. Berkes, M. Franzius, H. Sprekeler, and N. Wilbert (2011). "Slow feature analysis". In: *Scholarpedia* 6.4, p. 5282. DOI: 10.4249/scholarpedia.5282.

Wiskott, Laurenz and Terrence J. Sejnowski (2002). "Slow Feature Analysis: Unsupervised Learning of Invariances". In: *Neural Computation* 14.4, pp. 715–770. DOI: 10.1162/089976602317318938.

Wyss, Reto, Peter König, and Paul F. M. J Verschure (Apr. 2006). "A Model of the Ventral Visual System Based on Temporal Stability and Local Memory". In: *PLoS Biology* 4, e120.

Yang, Shichao and Sebastian Scherer (2019). "CubeSLAM: Monocular 3-D Object SLAM". In: *IEEE Transactions on Robotics*, pp. 1–14. DOI: 10.1109/TRO.2019.2909168.

Zhang, Zichao, Christian Forster, and Davide Scaramuzza (2017). "Active exposure control for robust visual odometry in HDR environments". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3894–3901. DOI: 10.1109/ICRA.2017.7989449.

Zhao, Zhongliang, Jose Carrera, Joel Niklaus, and Torsten Braun (2018). "Machine Learning-Based Real-Time Indoor Landmark Localization". In: *Wired/Wireless Internet Communications*. Springer International Publishing, pp. 95–106. ISBN: 978-3-030-02931-9.

Zhu, Kaiying, Xiaoyan Jiang, Zhijun Fang, Yongbin Gao, Hamido Fujita, and Jenq-Neng Hwang (2021). "Photometric Transfer for Direct Visual Odometry". In: *Know.-Based Syst.* DOI: 10.1016/j.knosys.2020.106671.

Zito, Tiziano, Niko Wilbert, Laurenz Wiskott, and Pietro Berkes (2009). "Modular toolkit for Data Processing (MDP): a Python data processing framework". In: *Front. Neuroinform.* 2.8. DOI: 10.3389/neuro.11.008.2008.