Learned Descriptors for Scalable and Efficient Visual Place Recognition

Saravanabalagi Ramachandran

A dissertation submitted for the degree of

Doctor of Philosophy

Department of Computer Science

Maynooth University

April 2025

Head of Department: Dr Aidan Mooney

Supervisor: Prof John McDonald

Abstract

Visual Place Recognition (VPR) is a task in computer vision that involves matching images of an environment to previously visited locations, enabling systems to identify and recognise places based on visual information. VPR has emerged as a widely studied topic in computer vision and mobile robotics, driven by its applications in autonomous navigation, image retrieval, and loop closure detection. Over the past decade, the field has witnessed significant progress, fuelled by improvements in camera hardware, the proliferation of mobile devices, and the growing availability of public image datasets. Researchers have increasingly utilised deep learning techniques to tackle the challenges of VPR, particularly those related to appearance changes and varying viewpoints that traditional descriptors struggled to address.

Despite these advancements, several interconnected challenges hinder the deployment of reliable and scalable VPR systems in automotive applications. Utilising large-scale sequential datasets poses significant difficulties due to diverse recording conventions, redundant visual content, and limited viewpoint variance, complicating training processes for deep learning. Additionally, efficiently categorising scenes without explicit object identification introduces considerable computational and methodological complexities. Furthermore, VPR systems face challenges related to scalability, primarily due to the computational demands associated with rapid retrieval of images for localisation along extensive trajectories spanning several kilometres. This thesis specifically addresses these critical challenges.

First, we introduce OdoViz, a comprehensive and unified framework designed for efficient dataset exploration, visualisation, analysis, curation, and preparation of bespoke training data from heterogeneous datasets. OdoViz streamlines the creation of standardised, tailored datasets essential for robust VPR model training.

Secondly, we elaborate on the development of robust learned image descriptors utilising large sequential datasets. We introduce a novel discretisation approach that segments trajectories into visually similar regions, facilitating efficient online sampling of triplets for contrastive learning. We present a detailed training regime involving tailored data subsets, a modified architecture, and a custom loss function for stable contrastive training, optimised to generate robust learned image representations.

Thirdly, we propose an efficient scene categorisation method leveraging Variational Autoencoders (VAEs). Our approach encodes images into compact, disentangled latent spaces without explicit object recognition, enabling rapid categorisation into urban, rural, and suburban contexts. This method achieves exceptional computational efficiency, with inference times under 100 μs, making it suitable for use as a pretext task in real-time automotive applications.

Finally, addressing the scalability concerns, we introduce a hierarchical framework utilising learned global descriptors to facilitate rapid retrieval over extensive distances while maintaining robust localisation performance. Through extensive experimentation, we identify continuity and distinctiveness as key properties of effective global descriptors for scalable hierarchical mapping, and propose a systematic method to quantify and compare these characteristics across various descriptor types. Our VAE-based scene descriptors achieve up to 9.5 x speedup on the longest evaluated track, St Lucia (17.6 km), while maintaining the same recall performance over longer trajectories, demonstrating their effectiveness in hierarchical localisation.

Together, these contributions address the identified VPR challenges, laying the groundwork for scalable and efficient VPR systems leveraging learned representations, suited for deployment in diverse real-world automotive environments.

Acknowledgements

I would like to express my gratitude to my supervisor, John McDonald, whose guidance, wisdom, and support have been instrumental in the development of my thesis and my evolution as a researcher. I would also like to extend my gratitude to Ganesh, Senthil, and Jonathan at Valeo, who gave useful insights and directions for my research.

Research presented in this thesis was supported by Science Foundation Ireland grant 13/RC/2094 to Lero, the Irish Software Research Centre, and grant 16/RI/3399. Additionally, this study was funded by the HEA COVID extension fund. These funds are gratefully acknowledged. I was fortunate to have met colleagues Toby, Louis, Will, Rob, James, Eduardo, Amit, Sutirtha, and Dharani, who have become dear friends along this journey. Those puzzle-solving sessions and lunchtime conversations ranging from scientific discussions to sharing life's little moments added spice to what could have otherwise been ordinary days. A special mention goes to Dharani for the countless late-night conversations about mathematics, statistics, and physics that have been both intellectually stimulating and personally enriching. I am further grateful to my friends and other wonderful people who brought excitement and joy into my life during this research journey.

Most importantly, I would like to express my gratitude to my mum and dad. Their enormous hearts that allowed me to study abroad, explore the world, and pursue my passions deserve more recognition than these words can convey. Despite being separated by oceans and continents, their boundless trust in my decisions and financial and emotional support, coupled with moral encouragement, have given me the freedom to chase my dreams. To my late dad, this work resonates with the love, support, and unwavering belief you always had in me.

Finally, a special thank you to my wife, Ruby, who has been my rock throughout everything. Her courageous decision to move to another country just to be by my side speaks volumes of dedication and love. Her presence has made this challenging journey not just bearable but beautiful.

In this journey that often felt solitary, I was never truly alone, thanks to all these wonderful people. While these words are printed in black and white, the memories and gratitude they represent are painted in the most vibrant colours of my life.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

Saravanabalagi Ramachandran.

Maynooth, Ireland,

April 2025.

Table of Contents

Chapter 1

Introduction

Visual Place Recognition (VPR) is the task in computer vision and mobile robotics of matching images of an environment to previously visited locations. At its core, VPR implicitly leverages visual cues such as distinct keypoints, patterns, objects, and landmarks within its detection and matching process. Unlike general image retrieval, VPR requires understanding the spatial and environmental context of images, which is crucial for applications like autonomous navigation where accurate place recognition is essential for mapping and localisation.

The complexity of VPR arises from the vast range of variations in the appearance of real-world places due to changes in lighting, weather, seasons, and viewpoints (see Figure 1.1 and Figure 1.2). The ability to reliably recognise places despite such variations makes VPR a particularly demanding problem. The past two decades have seen significant progress in the development of robust VPR techniques, driven by the necessity to address these challenges [1, 2]. These resulting techniques are now central to the development of dependable and resilient autonomous systems in domains ranging from self-driving vehicles to augmented and virtual reality headsets.

Within the domain of SLAM, VPR plays a particularly important role, providing global constraints on the robot's trajectory estimates and enhancing the accuracy and reliability of the mapping and localisation process. As a robot autonomously traverses the environment, it continuously captures data about its surroundings through cameras and other sensors. Fusing data from its sensors, the robot builds a map of its environment while also estimating its pose. When operating over large scales, the solution tends to

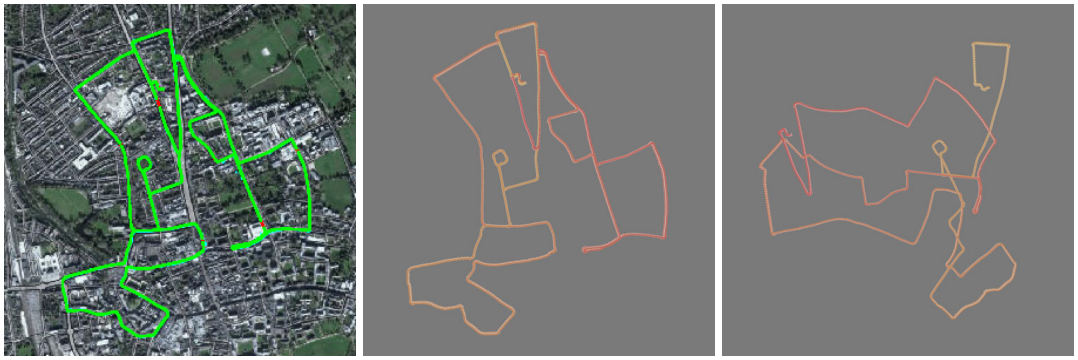
Figure 1.1: Images of the same place taken at different seasons highlighting the challenge of place recognition in the context of extreme changes in visual appearance. Image Credits: visuallocalization.net

Figure 1.2: Images of the same place taken in all four seasons from the Nordland dataset [3]

diverge in an unbounded fashion as a result of the accumulation of small errors (*a.k.a.*, drift) in the estimates of the robot's pose; see Figure 1.3.

To address this problem, the VPR module continuously attempts to match its current sensor data to previously visited locations. The resulting *loop closures* provide an important constraint by estimating the globally accumulated error in the robot's pose. Correcting for this error helps ensure the new sensor data is aligned accurately with the pre-existing map and that the resultant solution is globally consistent. Without loop closures, visual SLAM reduces to odometry, leading to the robot interpreting an infinite world, exploring new areas indefinitely [4, 5]. Furthermore, false negative matches will result in delayed correction of accumulated errors, while false positives will result in incorrectly merging regions, typically resulting in catastrophic failure of the system [6]. Reliable loop closing is hence both essential and hard. SLAM solutions incorporating VPR have direct applications in building better and more reliable autonomous navigation for self-driving cars and Unmanned Ground Vehicles (UGVs), autonomous drones or Unmanned Aerial Vehicles (UAVs), and other mobile robotics applications in both indoor and outdoor settings; see Figure 1.4.

Maps constructed through the measurement of local pose changes derived from an Inertial Navigation System (INS) or Visual Odometry (VO) alone exhibit high accuracy over short distances but experience drift over larger scales, accumulating significant



(a) Satellite Imagery (b) GPS (c) VO

Figure 1.3: (a) Trajectory in the Oxford RobotCar dataset [7] overlaid on Satellite Imagery data (b) Coloured GPS observations showing start (red) and end (yellow) (c) Map built only using relative pose values derived from Visual Odometry (VO), which is not globally accurate. Notice the disconnected start and end locations mapped far away from each other, which are indeed the same. Best viewed zoomed in colour on a computer screen.

Figure 1.4: Images showing various applications that utilise VPR systems for navigation. (Top Left) A Honda CR-V fitted with comma.ai openpilot. (Top Right). South Korean Team KAIST's robot turning a valve in the DARPA Robotics Challenge. (Bottom Left) Boston Dynamics Robot Spot climbing the stairs. (Bottom Right) An autonomous sprayer drone from DJI used in agriculture.

Figure 1.5: Left: Mapping error (red) caused by poor GPS reception in Oxford, where raw GPS values are shown in cyan. Right: A loss of reception causing significant positioning error over a large portion of the route [7]. Best viewed zoomed in colour on a computer screen.

errors; see Figure 1.3. When the Global Positioning System (GPS) measurements are combined with INS data, the mapping system benefits from the high-rate, precise INS measurements for short-term navigation and the global positioning accuracy of GPS for long-term trajectory and map corrections. Thus, when reliable location estimates are available as the robot navigates, more accurate maps can be built.

Whilst there have been numerous advancements to increase the precision of GPS estimates, the Estimated Position Error (EPE) can be significantly elevated — in the order of hundreds of metres, especially when the GPS receiver has limited visibility to satellites, for example, when driving a car in a tunnel, heavily forested areas, or in urban canyons — resulting in a highly inaccurate estimation of location, rendering it unsuitable for practical use in such situations, as shown in Figure 1.5. Although the system can be quickly recovered from such situations once better reception is available, the problem of robot navigation for indoor, marine, underground, and extraterrestrial applications that cannot leverage GPS positioning remains. Additionally, solving the place recognition problem using visual cues contributes valuable vision intelligence and transferable knowledge, which can prove useful for many tasks that rely on vision.

Early VPR approaches to recognising places relied on the detection and description of corners, or keypoints, in the query image and matching them with a collection of

reference images to find the closest match. While overall variation in brightness can be modelled using gradients, further variations in scale and rotation can be accounted for through a myriad of invariant feature descriptors, such as SIFT [8], SURF [9], BRIEF [10], ORB [11], and AKAZE [12]. These invariant feature descriptors enable finding matches between images captured within small time intervals where the images exhibit slight viewpoint shifts, small illumination changes, etc. As these local feature descriptors describe only a keypoint or a salient local region in an image, they are limited to only comparing regions of the image.

Global descriptors, on the other hand, encapsulate the characteristics of an image into a single, unified vector, providing a representation for the image, describing its overall structure and content. Local feature descriptors can be aggregated to form a global feature representation using approaches like Bag of Visual Words (BoVW) [13, 14, 15], Vector of Locally Aggregated Descriptors (VLAD) [16], and Fisher Vector (FV) [17].

Conversely, global representations that directly describe images holistically, such as Histogram of Oriented Gradients (HOG) [18] and GIST [19], can also be utilised. Adopting a holistic approach to building global feature descriptors is more efficient for many applications. This efficiency is particularly advantageous in tasks such as object recognition, scene categorisation, and VPR systems, where the ability to match images quickly and accurately is crucial.

Although image representations based on handcrafted feature descriptors have been demonstrated under mild perceptual changes, they are not as robust for matching images with challenging illumination, weather, seasonal, and viewpoint changes [20], such as those shown in Figure 1.1 and Figure 1.2. For VPR, this necessitates the ability to match images despite drastic appearance changes caused by seasons, such as snow on lawns, trees, and roads, and other changes in the environment, including those affecting buildings and landmarks. Recent advancements in deep learning have revolutionised this aspect, enabling the creation of learned descriptors that can capture the salient elements of the images more comprehensively [21].

As such, learned image descriptors represent a paradigm shift in the field of computer vision by enabling the extraction of a global descriptor directly from the entirety of an image, eschewing the traditional reliance on predefined interest points. Learned descriptors seek to capture the global characteristics of the image in a single, holistic

representation in a manner that is both robust and representative of the overall visual appearance. The synthesis of detection and description into a singular framework marks a significant advancement in computational efficiency and effectiveness within both VPR approaches and the broader field of image analysis. This exemplifies the overarching trend that scalable, data-driven methods often surpass traditional handcrafted solutions [22].

In addition to identifying the fine and distinct features within an image using the image descriptor, the capability to comprehend the configuration of spaces and categorise the broader scene also plays a crucial role in navigation and mapping tasks. Scene categorisation is a core computer vision task that classifies images into predefined categories (e.g., beach, restaurant, or mall) by analysing their overall content, objects, and spatial layout. Scene categorisation provides contextual information about the scene, enabling VPR systems to draw higher-level insights about the surrounding environment. As such, this reasoning about complex and diverse environments to obtain contextual cues is essential for intelligent systems to predict and interpret ongoing or future events.

Furthermore, such techniques that integrate scene categorisation contribute to the development of efficient hierarchical mapping and localisation methodologies. For instance, in retrieval tasks, extensive areas of scenes that bear no relevance to the anchor image can be efficiently bypassed. Such an approach not only streamlines the process of mapping and localisation but also enhances the overall efficiency and effectiveness of VPR systems in navigating and understanding diverse environments.

1.1 Challenges

In this thesis, we address the following challenges associated with visual place recognition.

C1: Processing public datasets for curating data subsets for training, validation, and testing: Data-driven methodologies achieve superior performance to handcrafted approaches by directly modelling the real-world variation exhibited in the large-scale image datasets. Consequently, this introduces a significant demand for such datasets to train, validate, and test the resulting models. Although the availability of several publicly accessible datasets serves to alleviate this difficulty, successfully utilising these resources

for VPR tasks requires a detailed understanding of the datasets' characteristics, such as the environmental conditions they represent, scene diversity, and temporal variations. Ensuring the relevance and comprehensiveness of the training data is essential, as they affect the system's robustness and generalisability of the learned techniques. Furthermore, curating and processing datasets involves analysing and interpreting a complex array of metadata, which is essential for filtering and preparing subsets of data tailored to the training requirements. These requirements pose a challenge, which is further exacerbated by the lack of interoperability among the Software Development Kits (SDKs) provided for interacting with these datasets. This incompatibility stems from a variety of factors, including, but not limited to, the disparate Application Programming Interface (API) designs that govern access to the datasets, the use of different programming languages across SDKs, and differences in the data capture and recording conventions used.

C2: Developing and refining learned representations for VPR under challenging conditions: The curated data serves as the foundation for constructing a model capable of generating robust image representations. Training such models involves an optimisation challenge with two distinct objectives. First, the model must remain invariant to variations in illumination (e.g., changes in time of day, seasons, and weather conditions) and dynamic environmental factors (e.g., vegetation changes and the presence of non-static objects such as pedestrians, vehicles, and trash bins). Second, it must retain the capacity to distinctly represent images from physically disparate locations, ensuring accurate place recognition and localisation. Using existing training methodologies to train a VPR model on large sequential image datasets from public sources that exhibit long-term changes presents two key challenges: (1) the sequential datasets often include redundant visual content, particularly in scenes where the vehicle is stationary, such as at intersections or roundabouts, leading to repetitive information; (2) there is limited viewpoint variance, especially on single carriageway roads, despite multiple traversals of the same route at different times. These two issues pose conflicting requirements: the need to reduce redundant data while simultaneously increasing data diversity to capture varying viewpoints. Furthermore, while pre-building unique positive and negative image pairs or triplets for each epoch in contrastive learning improves generalisability, this approach becomes impractical with large datasets, necessitating an alternative method that can generate data batches on the fly during training. Additional issues include slow loss convergence, unstable training processes, and the risk of em-

bedding space collapse or explosion. The challenge, therefore, is to develop a model to produce robust embeddings that abstract away transient and redundant elements of the scene while preserving stable, unique features essential for accurate place recognition, all while addressing the complexities of training on large sequential datasets.

C3: The need for efficiently determining scene type to provide context for VPR tasks: In the domain of autonomous navigation, scene categorisation serves as an important building block required to augment capabilities in context-aware object detection, action recognition, and comprehensive scene understanding. This context provides a prior for various computer vision tasks, facilitating parametrisation of downstream processing tasks. In an automotive context, the ability to automatically differentiate between rural, urban, and suburban settings allows tuning of algorithms to the specifics of the environment, such as adjusting pedestrian detection thresholds, thereby improving performance and reliability. However, unlike object recognition, scene categorisation poses unique challenges, as images from different classes often share objects, textures, and backgrounds, resulting in visual similarity and ambiguity among categories. Although an alternative approach is to rely on GPS data for contextual cues such as determining city limits for heightened pedestrian detection, such approaches suffer from limitations. In particular, it requires a priori labelling of the environment, thereby necessitating label management and updating to cater for rapid and dynamic development around cities and suburban regions. In contrast, determining the scene type in real-time using on-board sensor measurements eliminates the need for external data sources. However, such a system must perform this task very efficiently and be capable of realtime operation to meet the demands of practical applications.

C4: Scaling VPR systems to long trajectories: As the scale of the operating environment of the robot increases, adopting a simple approach of exhaustively searching the database for potential matches results in a linear increase in complexity. Several studies [13, 23] have addressed this issue using indexing techniques, enabling modern approaches to scale. However, indexing requires additional storage space and can lead to increased maintenance overhead, as the index must be updated with every data modification. This necessitates a hierarchical approach that overcomes this limitation by organising images into a structured format, facilitating the rapid narrowing down of potential matches. The consequent search space complexity reduction through the use of hierarchy is crucial for enhancing the efficiency of place recognition tasks on trajectories that span several kilometres, especially in applications requiring real-time

or near-real-time performance, such as autonomous driving and robotic navigation.

1.2 Contributions

This thesis addresses the challenges discussed in the previous section through the use of learned representations for visual place recognition, proposing novel solutions and methodologies to advance the field. Our contributions are delineated as follows:

- In response to Challenge C1, which underscores the difficulties in harnessing publicly available datasets, we introduce OdoViz [24], a novel unified framework to facilitate the efficient exploration, visualisation, curation, sampling, and preparation of datasets and subsets of data from a wide set of public datasets for VPR research. By enhancing the usability of extensive datasets and facilitating features to derive standardised data subsets tailored to bespoke requirements, OdoViz serves as a foundational element for developing robust learned models in VPR.
- Addressing Challenge C2, we detail the development and training of VPR models that generate robust learned embeddings from images using contrastive losses utilising large sequential public datasets. We propose a novel approach of discretising trajectories into locations (or regions) containing similar images, allowing for efficiently sampling and obtaining of unique triplets during training. We employ adaptations to the loss function, architecture, and learning rate amenable to better loss convergence and to prevent training failures. We aggregate discretised locations and additionally utilise data augmentation techniques to add viewpoint variance. We efficiently construct training data batches in an online fashion, eliminating the need to pre-select and prepare triplets before each training epoch, a process that is computationally expensive. We train our model using online batches with progressively increasing difficulty by dynamically selecting challenging samples from strategically chosen locations during training. Our model successfully trains and produces embeddings that demonstrate improved retrieval performance on data with challenging conditions such as seasonal changes and day-night variations.
- For Challenge C3, we present a novel, highly compute-efficient, deep learning-based approach to scene categorisation utilising Variational Autoencoders (VAEs).

This approach employs a convolutional VAE to encode images into a multi-dimensional latent space without explicit object recognition. We train the VAE in an unsupervised manner on the image reconstruction task utilising large sequential datasets in order to capture high-level scene information. We propose to use the disentangled latent features from the encoder as compact, interpretable global features. We map these features to three scene categories: rural, urban, and suburban, using a light supervised classification head requiring fewer than 500 labelled images. With an inference time of only 60 μs on a consumer-grade desktop with an i9-9900K and NVIDIA 2080 Ti, our method efficiently categorises scenes.

- In tackling Challenge C4, related to the scalability and efficiency of VPR systems, we propose the use of compact learned global descriptors in hierarchical topological mapping. This approach aggregates similar images into location nodes using a learned global descriptor, dramatically enhancing the retrieval process's speed and efficiency. Through empirical analysis, we identify and define the characteristics of an ideal global descriptor supporting hierarchical matching amenable to scalable and efficient visual localisation and present a methodology for quantifying and contrasting these characteristics. We conduct a comprehensive evaluation of various global descriptors, identifying those that best support scalable and efficient hierarchical matching. The image representations we developed for scene categorisation emerge as particularly effective while maintaining the same recall performance in longer trajectories, demonstrating their utility in hierarchical systems.

1.3 Outline

The thesis is organised as follows:

In the Background chapter, we discuss various solutions proposed to visual place recognition, beginning with traditional approaches that focus on aggregating local features to recent learned approaches incorporating semantic, geometric, and topological information from the scene. We further explore scene categorisation techniques and their application in scalable and efficient hierarchical mapping systems.

In Chapter 3, we provide a comprehensive overview of the foundational elements of

VPR systems, presenting a thorough examination of the central elements of this research area. We elaborate on the various datasets that serve as benchmarks for guiding research and development and providing an objective measure of progress within the domain. This is followed by an analysis of existing tools developed for the processing, analysis, and visualisation of these datasets. We further expand on Challenge C1 pertaining to utilising these tools and datasets to prepare new subsets of data required for VPR research. We present a new tool, OdoViz, dedicated to odometry visualisation and processing, involving techniques that aid in curating data for training VPR models. We then highlight various metrics to measure the effectiveness, reliability, and robustness of different VPR approaches.

In Chapter 4, utilising datasets created using OdoViz, we explore and evaluate techniques to generate robust image embeddings addressing objectives mentioned in C2. We elaborate on building, customising, and tuning data-driven models with various techniques utilising contrastive learning to output embeddings that implicitly encode information amenable to place recognition in challenging conditions that involve moderate to extreme day-night, weather, and seasonal appearance changes. We elaborate on our novel approach of discretising trajectories into locations containing similar images to efficiently obtain triplets. We then detail the set of techniques and adaptations proposed to successfully train a VPR model with triplet loss using weakly supervised data curated from the Oxford RobotCar dataset [7], a large public dataset with over 100 traversals of the same route over a period of more than one year.

In Chapter 5, tackling problems posed by Challenge C3, we present a new deep learning based holistic global descriptor approach utilising VAE that encodes high-level scene information in a multi-dimensional latent space without explicitly recognising objects, their semantics, or capturing fine details. We discuss training the VAE in an unsupervised fashion on the image reconstruction task and use disentangled latent variables as global feature descriptors. This is followed by utilising a lightweight supervised classification head to map these features to the three scene categories: rural, urban, and suburban. We then present scene categorisation results and show that our approach is fast for realtime inference and efficient with a compact embedding size, suitable for use as a pretext task in autonomous vehicles.

In Chapter 6, we propose to use compact learned global descriptors in hierarchical topological mapping of environments to aggregate sequences of images with similar

appearance into location nodes, addressing scalability issues described in Challenge C4. While many learned descriptors with improved retrieval accuracy have been incorporated into place recognition methods to enhance overall recall, we instead focus on addressing the challenges of scalability and efficiency, in particular, when such methods are used on longer trajectories. We elaborate on identifying and defining the characteristics of an ideal global descriptor supporting hierarchical matching amenable to scalable and efficient visual localisation through empirical analysis. As part of this, we also present a methodology for quantifying and contrasting these characteristics. We then propose the use of compact learned scene descriptors that excel in continuity and distinctiveness characteristics as an efficient and scalable means for hierarchical topological mapping.

Finally, in Chapter 7, we summarise the contributions of the research presented in the previous chapters and discuss potential future directions.

Chapter 2

Background

Visual Place Recognition (VPR) systems enable a camera-equipped device to recognise previously visited places by comparing the visual information captured within images. Cameras, being a dominant sensor for perception, provide a rich source of data to carry out a variety of navigation-related tasks. The development of effective image descriptors is thus critical to VPR systems to facilitate fast, robust, and reliable loop closure detection. The problem of VPR has been a topic of research focus within robotics for more than two decades, during which time there have been a number of technical advances [2].

In this chapter, we introduce basic concepts, terminologies, and methodologies essential for understanding the subsequent discussions in this thesis. We begin by exploring what are now considered traditional approaches to the problem based around handcrafted features. We then cover data-driven approaches, where deep learning techniques leverage geometric, semantic, and temporal information to improve the robustness and efficiency of VPR systems. We further highlight the works that are particularly relevant to addressing the challenges of scalably and efficiently recognising places under varying conditions.

Limitations: We note that the structure of the review is designed to provide a representative and coherent narrative leading up to the research questions of this thesis, rather than to offer an exhaustive survey. Consequently, particular emphasis has been placed on approaches and concepts, including learned descriptors and hierarchical techniques, most pertinent to the scalability, efficiency, and robustness challenges

addressed in subsequent chapters.

2.1 Bag of Visual Words Descriptors

Early approaches to image retrieval were heavily influenced by the Bag of Words (BoW) model, originally developed for text-based document retrieval tasks [25]. In what has become a seminal paper in the field, Sivic and Zisserman adapted the technique to visual data, leading to the development of the Bag of Visual Words (BoVW) approach [13, 14, 15]. In the BoVW method, local feature descriptors are first extracted from a large set of training images and subsequently clustered using the k-means algorithm. The resulting cluster centres, or means, are treated as feature codewords, collectively forming a codebook of length k. Each image is then encoded as a compact k-dimensional vector, where the coefficient of the i^{th} dimension represents the number of descriptors that correspond to the i^{th} cluster. This encoding effectively generates a histogram that records the distribution of local feature descriptors across the k clusters. The overall concept is explained visually in Figure 2.1.

In earlier works, local feature descriptors like SIFT [8] and SURF [9] were used to construct the reference codebook from a training set during an offline phase. In subsequent years, the introduction of binary descriptors, such as BRIEF [10], BRISK [26], ORB [11], FREAK [27], and LDB [28], enabled faster processing and comparison. Furthermore, [29, 30, 31] bypassed the initial training step and built the codebook in an incremental manner as the robot explored the environment, allowing for online applications without the need for pre-built dictionaries.

Image retrieval in this context involves calculating distances between the query image's BoVW representation and those of the images in the search space, with a threshold determining the number of similar images retrieved. This approach was utilised in early Content-Based Image Retrieval (CBIR) systems, where images were represented as vectors encapsulating feature statistics [32]. This method was adapted for SLAM systems by continually adding the BoVW representation of the captured images to a database and querying for similar representations to identify and integrate loop closures as the system progresses.

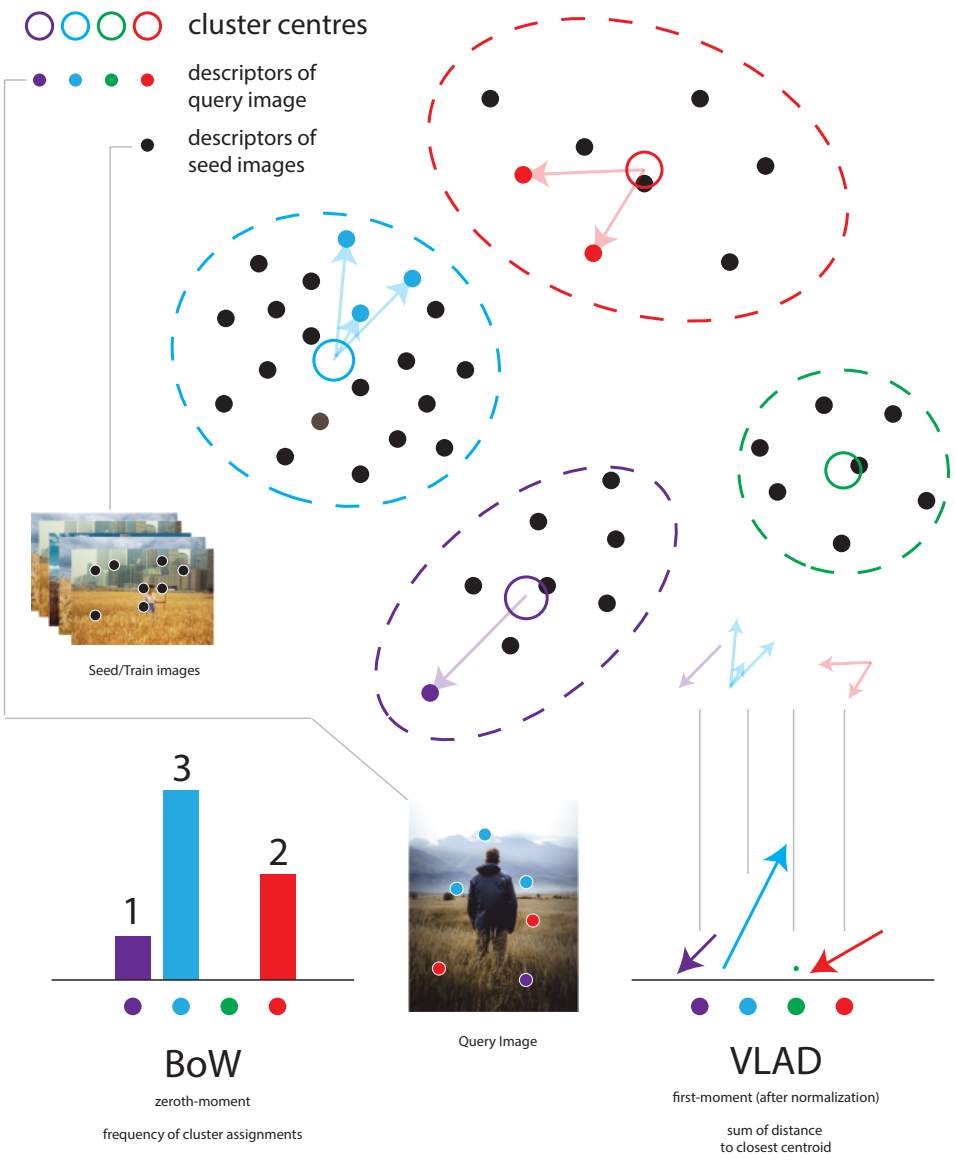


Figure 2.1: Illustration of the BoVW and VLAD concepts and the key differences between them. VLAD uses more space, a 128-d vector (representing aggregated residuals) in place of a single number (frequency) for each centroid. However, incorporating first-order feature-codeword statistics provides more distinctive information to classifiers, resulting in improved performance when compared to the BoVW image representation [16].

2.1.1 TF-IDF and Inverted Index

In Content-Based Image Retrieval (CBIR) systems, the discriminative power of visual features (words) varies, much like terms in text retrieval systems. The Term Frequency-Inverse Document Frequency (TF-IDF) framework, widely used in text retrieval, was adapted and applied to the image domain in [13]. In this framework, images are represented using a BoVW vector with TF-IDF weighted feature frequencies, marking a shift from using raw histogram counts. This weighting scheme promotes features that occur more frequently within the given image and, at the same time, diminishes features that occur frequently across the training set. For visual data, TF-IDF score is given by:

$$\text{TF-IDF}(f, I) = \text{TF}(f, I) \times \text{IDF}(f) \tag{2.1}$$

$$\text{IDF}(f) = \log\left(\frac{N}{n_f}\right) \tag{2.2}$$

where,

I = image
f = local feature descriptor (or visual word)
N = total number of images
n_f = number of images containing f
$\text{TF}(f, I)$ = term frequency, frequency of f in I
$\text{IDF}(f)$ = inverse document frequency

As such, this approach refines the traditional histogram-based comparison by incorporating TF-IDF scores into the retrieval process, allowing for more discriminative matching of images in large datasets. To enhance retrieval efficiency, inverted indices were employed, mapping local features (words) to all the images (documents) in which they appear. As local feature descriptors are computed, the inverted index is updated for each feature codeword in the codebook. When matching a new image, the system only compares it to images sharing similar features, significantly speeding up the retrieval process.

Further to the above, a stop-list, which excludes the top 10% and the bottom 5% of the descriptors found in the corpus of initial images, was used to reduce the number of mismatches and the size of the inverted file while maintaining a sufficient visual vocabulary. This process removes the most frequent visual words that occur ubiquitously

in most images. These visual words generally lack distinguishing characteristics, such as a patch of sky and a patch of road, analogous to the common words like *is* and *the* in text-based information retrieval. Similarly, the least frequent visual words, which can contain particularly unique patterns or objects that occur only once or twice in the entire set, are also eliminated, as they do not prove to be helpful in retrieving images. The performance of this technique, however, degrades as the database grows, making it more time-consuming to update the inverted index.

Building on the work in [13], [23] introduced a hierarchical approach that organises the visual vocabulary into a vocabulary tree utilising a tree structure. This approach supports much larger vocabularies, improving retrieval rates by significantly reducing the number of images considered during each query. Consequently, this method allows for efficient searches across databases containing millions of images.

2.1.2 VLAD and FV

The BoVW representations often result in sparse encoded vectors, as new images typically contain descriptors corresponding to only a subset of the k clusters. Selecting a smaller k can lead to underfitting, limiting the discriminative power of the codebook, while a larger k may cause overfitting, resulting in very sparse high-dimensional descriptors. Addressing this issue, advanced encoding methods such as the Vector of Locally Aggregated Descriptors (VLAD) [16] and the Fisher Vector (FV) [17] store additional statistics between codewords and local feature descriptors.

VLAD [16] extends the BoVW approach, where for each of the k clusters, the residuals (vector differences between descriptors and their closest cluster centres) of image descriptors are accumulated. The 128-D[1] sums of residuals for the k clusters are concatenated into a single $k \times 128$ dimensional descriptor, which is then L_2 normalised. Thus, we store first-order feature-codeword statistics (i.e., the sum of the difference between the descriptor and the mean of the corresponding word's cluster) in the VLAD vector. Figure 2.1 illustrates constructing a VLAD vector for a given query image.

Building on this work, [33, 34] employed soft cluster assignment by assigning each descriptor multiple centroids weighted by their distance from the descriptor. The

[1] assuming 128-dimensional feature descriptors are used; e.g., SIFT.

consequent richer representation of feature statistics and the association with multiple feature codewords resulted in improved performance in object categorisation and video retrieval.

The MultiVLAD method proposed in [35], utilises multiple spatial VLAD representations to enable the retrieval and localisation of objects that only occupy a small portion of the image. The image is tiled, and multiple VLAD descriptors are generated at different scales to overcome the inferior small object retrieval performance of VLAD compared to BoVW. Furthermore, an L_2 intra-normalisation scheme was developed for VLAD that addresses the problem of burstiness, where a few large components of the VLAD vector can adversely dominate the similarity computed between VLAD representations. With this approach, each of the aggregate residual vectors (128-D) corresponding to one of the k clusters is L_2 normalised. This method demonstrated improved retrieval performance over non-normalised and L_2 normalisation applied to the whole vector.

The Fisher Vector [17, 36], a special and improved case of the general Fisher kernel [37], represents an image by its deviation from a generative model, typically a Gaussian Mixture Model (GMM). By encoding the gradients of the log-likelihood with respect to the GMM parameters, FV captures richer information about the distribution of local features, including their mean and variance. Thus, FV encodes the first- and second-order feature-codeword statistics in the image representation. A comparative study of local feature representations reveals that VLAD and FV were found to perform almost equally well, with FV performing slightly better for larger codebooks [38].

2.1.3 SLAM Implementations

Building on earlier work applying BoW models to images, particularly within the context of loop closure in SLAM, FAB-MAP [39] presents a learned generative model for the BoW data and defines a probabilistic approach over the BoW representation. FAB-MAP 2.0 [39] extends this framework by incorporating improvements such as optimised Cholesky decomposition for faster inference, better scalability, and increased accuracy. DBoW [40] extended this framework for real-time operation by employing a combination of FAST keypoints, BRIEF descriptors, and a hierarchical tree-based vocabulary, combined with an inverted file structure for efficient queries. In all these

methods, a global BoVW-based image descriptor is computed by aggregating local features.

To address the challenges posed by illumination variations within environments and the resulting difficulties in matching images using keypoint-based descriptors, [41] introduced the concept of Illumination Invariant Imaging. The approach proposes a transformation into a colour space that is invariant to changes in illumination, thereby yielding feature maps robust to alterations induced by different times of the day, shadows, and lighting conditions. This method enhances the robustness of image matching processes by significantly reducing the impact of environmental lighting variations, facilitating more consistent and reliable recognition of visual features across diverse lighting scenarios.

[42] explores the use of binary descriptors for sequences of images to form binary codes combined with illumination-invariant techniques described in [41], using an efficient Fast Library for Approximate Nearest Neighbours (FLANN)-based matches to measure similarity between image sequences. The technique, named ABLE-M (Able for Binary-appearance Loop-closure Evaluation — Monocular), uses a binary description and matching method to provide a significant reduction in memory and computational costs, which is necessary for long-term performance. ABLE-M outperformed WI-SURF [43], BRIEF-Gist [44], FAB-MAP [39] and SeqSLAM [45] evaluated on the St Lucia dataset (along the day) [46], Alderley dataset (along the day and night) [45], CMU-CVG Visual Localisation dataset (along the months) [43], Nordland (along the seasons) [3, 47].

Although the advances provided by the BoVW and its extensions permitted a reliance on VPR within SLAM systems, the approach lacks the repeatability and robustness required to deal with the challenging variability that occurs in natural scenes caused by different times of the day, weather, lighting, and seasons, as shown in Figure 1.1.

2.2 Handcrafted Holistic Descriptors

Global descriptors characterise an image holistically by processing the entire image to produce a singular description. Histograms such as colour histograms or histograms

of oriented gradients (HOG) [48, 49, 18] provide a compact way of representing an image and are generally fast to compute. The Pyramid Histogram of Oriented Gradients (PHOG) [50], an enhancement of the original model, computes histograms for oriented gradients across different sub-regions of each image.

The Gist descriptor was introduced in [19] as one of the earliest holistic representations designed to capture the dominant spatial structure of a scene. By applying a series of Gabor filters at multiple scales and orientations across the entire image, the Gist descriptor extracts information related to the spatial frequency content of the scene and represents the global arrangement of spatial features rather than focussing on local details. Although initially developed for scene recognition, it has been foundational in various image processing applications, and its capacity to efficiently represent scene configurations led to researchers integrating it with other description techniques.

Building on the efficiency of the BRIEF binary descriptor [10], BRIEF-Gist [44] was proposed to enable fast holistic image description with reduced computational overhead. In this method, the image is first downsampled to a suitable small patch size (e.g., 60 x 60 pixels), and then the BRIEF descriptor is computed around the centre of the downsampled image. Alternatively, the image can be divided into patches, and the BRIEF description of each patch can be stacked to obtain the global descriptor. This description technique, thus, allows for a very simplistic appearance-based representation for use in a place recognition system. Similarly, WI-SURF [43] extended the conventional SURF approach, which typically extracts features from key points, to use the entire image.

SeqSLAM [45] was developed as a sequence-based algorithm that compared brightness patterns, where it searches for optimal matches within local image sequences. As such, SeqSLAM does not rely on keypoints or feature extraction methods (like SIFT, SURF, or ORB) that are common in other image matching and computer vision tasks and relies on finding coherent sequences of downsampled image matches instead.

Later, optical flow-based descriptors such as Optical Flow Moment (OFM) and Optical Flow Shape Context (OFSC) [51] were introduced to incorporate motion cues. These descriptors capitalise on the dynamic changes captured by optical flow fields, incorporating statistical attributes from the flow to uniquely define each location.

Although many of these descriptors have been utilised in various forms of mapping,

they are less robust to occlusions, strong illumination changes, and variations in viewpoint, leading to diminished discriminative capability, especially when compared to the data-driven approaches introduced over the last decade.

2.3 Learned Descriptors

In the domain of learned image descriptors, two principal backbone architectures have emerged for feature extraction: Convolutional Neural Networks (CNNs) [52, 53] and transformers. CNNs extract hierarchical features from images using convolutional layers, progressively capturing local and global patterns. Transformers, on the other hand, divide images into patches and utilise self-attention mechanisms to capture both local features and long-range dependencies across the entire image. In both architectures, feature extraction is typically followed by pooling or aggregation layers, such as max pooling and average pooling, or methods like VLAD. These processes condense the extracted feature maps or patch descriptors into compact global descriptors, which serve as efficient, robust representations for image recognition and other downstream tasks.

Following the compelling results of CNNs over traditional methods in tasks such as object classification and recognition [54], semantic segmentation [55] and feature learning [56], researchers have increasingly incorporated semantics and geometry in addressing the problem of place recognition in challenging conditions. However, original CNN architectures, such as AlexNet [54], VGGNet [57], and ResNet [58], proposed for visual tasks such as object recognition, are not directly suitable for place recognition tasks. Unlike object recognition, which focuses on identifying individual objects, place recognition requires encoding the broader spatial layout and complex relationships between elements such as buildings, roads, and their relative positions within a scene, often under varying appearance conditions and viewpoints. Moreover, many *off-the-shelf* techniques limit building convolutional networks tailored to solving specific tasks such as object classification and semantic segmentation in an end-to-end fashion.

Supervised learning requires tens of thousands of labelled training samples. Although it has seen significant success in recent years, the requirement for manual labelling of such large datasets has served as a significant barrier to progress. For tasks

like dense semantic segmentation, it is often impractical to account for every object and to encompass every pixel associated with each object. This task becomes significantly more time-consuming, as it can take several minutes to label even a single image. To make the network learn semantics, geometry, illumination, colour, weather, objects, vegetation, and so on, given the amount of time and tedious manual effort required to label these features, it is infeasible to label these individually for hundreds of thousands of images.

2.3.1 Weakly Supervised Techniques

Weakly supervised learning is a machine learning framework where the model is trained using examples that are only partially annotated or labelled. Unlike supervised training, for example, in image recognition tasks, where mapping each image to one of the definitive classes is a necessity, in weakly supervised training, it is sufficient to have image correspondences to group similar images into pairs, triplets, or quadruplets. This is well suited for problems like place recognition, where it is not possible to classify images under a set of predefined locations.

Distance metric learning (DML) is a crucial technique in image representation learning, aimed at optimising the metric used for assessing similarities between images to support various image-related tasks, including classification, retrieval, and clustering [59, 60, 61]. The objective of DML is to learn a metric that reduces the distances between similar images and increases the distances between dissimilar ones, thereby aligning the metric more closely with semantic similarities. This objective is often achieved by transforming the feature space to make distances in the transformed space reflect the true categories or labels of the images. Deep learning approaches utilise a range of architectures, including CNNs, fully connected layers, and transformers, to apply complex, non-linear transformations for feature extraction and metric optimisation. Such neural networks can be trained using various contrastive losses [62, 63, 64].

FaceNet [63] provided one of the earliest approaches in this area, proposing the use of a triplet loss to train a CNN on a dataset of human faces, which was then made to output embeddings for face recognition using image retrieval. Building on top of this idea, Person ReID [65] proposed a method using batch hard and batch all mining strategies. Quadruplet Loss [64] further extends this idea to train using quadruplets

instead of triplets with better retrieval for newly learnt classes (classes that are not present in the training set), demonstrating an improvement in generalisability.

Adapting such contrastive learning techniques for VPR, [66] introduced a method to localise a ground vehicle using publicly available satellite imagery as the only prior knowledge of the environment. This method employs a Siamese CNN to produce embeddings that are robust to viewpoint and appearance variations and utilises a particle filter to eliminate false positives. To train the neural network to generate embeddings, the pairwise contrastive loss (explained later in Section 2.3.1) between the embeddings of the image captured from the front camera and its corresponding satellite image is minimised, updating weights of both branches of the Siamese network independently.

The most notable learned image descriptor technique utilising a contrastive approach is NetVLAD [67], which reformulated VLAD through the use of a deep learning architecture, resulting in a CNN-based feature extractor that utilises weak supervision to learn a distance metric based on the triplet loss. In this method, a VGG [57] based CNN is employed to extract features utilising a generalised differentiable VLAD aggregation layer with a soft cluster assignment for end-to-end training. Subsequent extensions to NetVLAD proposed over the last few years, such as [68, 69, 70, 71], produce patch-level features and/or capture multi-scale features, demonstrating superior image retrieval performance.

There have also been improvements to the loss function to enhance the performance. For example, [72] introduced a new learning strategy to learn a large margin in a multi-stage manner while making the learned features more discriminative by exploiting multiple levels of feature maps. [73] developed an end-to-end top-k precision optimisable deep neural network by sampling misplaced images along the top-k nearest neighbour boundary for the loss signal. Consequently, several successful visual geolocalisation approaches [67, 74, 69, 70, 75, 76] have adopted contrastive loss as a critical technique, often employing a triplet loss that mainly relies on the mining of negative examples across the training database.

More recently, CosPlace [77] was introduced for visual geo-localisation, dispensing with the typical contrastive learning approach that relies on mining negative examples. Inspired by CosFace [78] and its implementation of the Large Margin Cosine Loss (LMCL), the training phase is reformulated as a classification task to address the

scalability limitations of previous methods. CosPlace employs a network architecture consisting of a conventional CNN backbone followed by Generalised Mean (GeM) pooling [79] and a fully connected layer with an output dimension of 512. CosPlace streamlines the process of learning from a large dataset without the need for explicit mining by splitting the database into classes based on GPS coordinates and headings and then training on groups of non-adjacent classes. Through these techniques, CosPlace enables more precise city-wide real-world visual geo-localisation by improving image retrieval using smaller descriptors.

In our work, we build upon a CNN architecture utilising triplet loss, introducing various adaptations to facilitate training on large sequential datasets that account for day-night and seasonal variations.

2.3.2 Sequence-based Techniques

Sequence-based VPR techniques exploit temporal and spatial continuity in image sequences, improving robustness and accuracy in identifying locations under varying environmental conditions and viewpoints. While descriptors incorporating temporal cues have been utilised within the broader field of computer vision [80, 81], only a handful of works have explicitly adopted it for VPR.

[82] proposed a method for robust visual localisation across seasons exploiting network flows to leverage sequential information to improve the localisation performance and to maintain several possible trajectory hypotheses in parallel. In this method, a semi-dense image description based on HOG features as well as global descriptors from deep CNNs pretrained on ImageNet is used for robust localisation.

[83] introduced an effective VPR method based on a multi-sequence map, employing a graph-based sequence-to-sequence localisation and a multi-trajectory place recognition. This approach demonstrated VPR against sequences from different sources: cars, bikes, street-view imagery from Google Street View, and YouTube videos without any constraints on shape, length, or visual change of the trajectories.

2.3.3 Semantics-aware Techniques

Semantic-aware VPR techniques enhance the discriminative capability of the models by incorporating context and object-level understanding, which potentially aids in achieving more accurate and stable recognition across diverse and dynamically changing environments.

In SegMatch [84], objects are segmented from LiDAR point cloud data accumulated for approximately one second as the vehicle traverses its trajectory, with the 3D object's description stored along with its semantic label. When a new frame arrives, moving objects are eliminated using their semantic labels, where the remaining objects are then compared to those from earlier segments using their 3D descriptors. [20] introduced an approach to semantics-aware visual localisation under challenging perceptual conditions, building dense saliency maps describing the scene with regions that are geometrically stable over large time periods. With this, a heat map was built, associating with each pixel the probabilities for that region to be geometrically stable.

Local Semantic Tensors (LoST) [85] used output tensors from one of the intermediate convolutional layers, *conv5*, of a modified version of the dense semantic segmentation neural network RefineNet [86]. These tensors were used to compute a deviation from the mean tensor for three semantic labels (i.e., road, building, and vegetation), which were then flattened to a vector representation. During the traversal, the images are mapped to embeddings and are compared with existing embeddings for potential matches. To avoid false matches as a result of a sudden change in a single frame, the mean embedding within a window of 15 frames centred around the associated frame is calculated. The images that have an embedding similarity greater than a defined threshold are further eliminated using keypoint correspondences by matching maximally activated regions in the feature maps to find the final loop closing candidate. The overall place recognition pipeline that uses both the LoST descriptor and the keypoint correspondence is referred to as LoST-X. LoST-X demonstrates double recall at 100% precision [85] compared to NetVLAD on the Oxford RobotCar dataset [7].

More recently, Semantic Reinforced Attention Learning Network (SRALNet) [76] introduced feature embeddings enhanced with task-relevant visual cues. This method utilises semantic priors and data-driven fine-tuning to refine its inferred attention. The network introduces an interpretable local weighting scheme designed to suppress mis-

leading features based on a hierarchical feature distribution, followed by a semantically constrained initialisation to reinforce the local attention with semantic priors.

Learned semantic techniques in VPR have demonstrated potential to enhance the accuracy of identifying locations by adding contextually rich, descriptive information to the visual cues extracted from images. However, these techniques often necessitate pixel-level dense annotations across extensive image datasets, which are both labour-intensive and expensive. Moreover, the dependency on densely annotated datasets restricts the training process to the specific images that have been annotated, thereby limiting both the scalability and generalisability of VPR models trained in such an environment.

2.3.4 Self Supervised and Unsupervised Techniques

In response to the challenges associated with manual dense data labelling, there has been a notable shift in recent years towards leveraging less restrictive forms of supervision. Researchers have increasingly adopted unsupervised and self-supervised learning paradigms to circumvent the limitations imposed by dense annotation requirements.

Autoencoders [87] laid the groundwork for learning compressed representations of data without labels by reconstructing the input image. The development of Variational Autoencoders (VAEs) [88] further advanced the field, particularly in applications like face generation, by learning to model the distribution of data in a latent space and generating new data samples from this learned distribution. Generative Adversarial Networks (GANs) [89] marked a turning point with their ability to generate high-quality images, outperforming other unsupervised generative techniques in terms of the visual fidelity of generated images.

At the same time, Self-Supervised Learning (SSL) began to gain traction and had been proven more effective in generating robust feature representations without the need for labelled data [90, 91]. SSL models are capable of learning useful representations from unlabelled data through contrastive learning or other mechanisms, such as using a *student-teacher* framework. Specifically, self-supervised learning approaches operate by creating labels on the fly for a predetermined pretext task, such as predicting parts of an image that have been intentionally obscured. This method makes efficient use of vast

amounts of unlabelled visual data to learn rich, generalisable feature representations applicable to tasks outside of the original pretext task. This approach helps to deepen the model's understanding of basic and intrinsic patterns in visual data, which is crucial for performing required downstream tasks such as object detection, semantic segmentation, and image classification.

Early SSL models learned rich data representations by engaging in tasks such as colourising images [92], predicting missing patches [93], estimating rotation angles [94], or solving jigsaw puzzles [95]. Although these tasks might appear straightforward to humans, they require a complex modelling of features, structure, and the occurrences of various visual elements in the real world. As such, the design of an effective pretext task often requires substantial domain knowledge, as the task must be sufficiently challenging and relevant to encourage the model to develop useful features. Accordingly, these pretext tasks are capable of effectively training the model to understand and interpret various types of data, including images, audio, and video [90]. The knowledge acquired through these pretext tasks is then applied to downstream tasks, where the previously learned representations are used to perform specific applications or solve particular problems. Over time, self-supervised learning techniques have increasingly narrowed the performance gap with supervised methods, as evidenced by their competitive results on ImageNet [96] and COCO [97] benchmarks for image recognition tasks. Noteworthy SSL techniques include Contrastive Predictive Coding (CPC) [98], Simple Framework for Contrastive Learning of Visual Representations (SimCLR) [99, 100], Momentum Contrast (MoCo) [101, 102, 103], and Bootstrap Your Own Latent (BYOL) [104]. CPC uses an autoregressive model to predict future representations in a latent space, capitalising on the inherent structure of data to learn without explicit labels. SimCLR advances this approach by using contrastive learning to enhance representation quality, effectively using data augmentation and a non-linear projection head to improve feature learning through maximising agreement between different augmented views of the same image. MoCo builds on these concepts by implementing a dynamic dictionary of samples and utilising a momentum encoder to ensure consistent representation. It introduces a queuing system to manage sample consistency and employs a momentum update strategy to stabilise the representations over time. BYOL, diverging from the reliance on negative pairs typically used in contrastive learning, employs a dual network architecture where the online network predicts the representation of a target network that is updated using a slow-moving average of the online parameters.

The advent of Vision Transformer (ViT)-based architectures like DINO v2 [105] further exemplifies this trend. These models utilise self-attention blocks within the transformer backbone to extract relevant features without explicit supervision. Very recently, AnyLoc[106] demonstrated a versatile VPR technique that works across a broad range of structured and unstructured environments without any retraining or fine-tuning. Features are extracted from intermediate layers across a pretrained DINO v2 ViT backbone, aggregated using a GeM or VLAD layer, and the global descriptors are projected using PCA. As such, this evolution towards scalable and broadly applicable models marks a significant advancement in the field of computer vision. Although transformers have proven to be highly effective for various computer vision tasks, their computational complexity remains a challenge, especially in resource-constrained environments such as for use in automotive applications.

In our work, we utilise a VAE-based architecture to learn robust scene features in an unsupervised manner from public datasets for scene categorisation in driving scenarios. Although VAEs may face challenges in reconstructing detailed features in images and can produce lower-quality reconstructions, we leverage their robust capability to capture the essential high-level features that are critical for effective scene categorisation.

2.4 Scalable and Efficient Approaches

Encoding an image as a single feature vector of predetermined dimension via the aforementioned learned global descriptor techniques necessitates conducting a comprehensive search across all encoded images to locate the nearest match for a query image using a similarity metric. Hence, the search duration increases linearly with the expansion of the image repository in the map. In the context of lifelong learning and multi-session SLAM, this linear increase in search time associated with loop closure detection can pose significant challenges. As the robot navigates and accumulates images over multiple sessions or extended periods, the resultant growth in the image database leads to progressively longer search times for loop closure detection. This scalability issue not only impairs the system's real-time operational capability but also limits the practicality of such systems in dynamic environments where efficient processing is paramount.

2.4.1 Indexing and Hierarchical Techniques

A crucial strategy for mitigating the time complexity associated with this process is the application of indexing techniques. The seminal work of [13] introduced an approach to search viewpoint invariant region descriptors using inverted file systems and document rankings similar to the ones used in text retrieval systems, as we described earlier in Section 2.1. Extending these foundational insights, [23] further refined the approach by developing indexing strategies for descriptors derived from local image regions.

The performance of the retrieval process can be markedly improved by employing selective search strategies based on specific scene categories. For instance, when presented with a query image featuring multiple high-rise buildings, it is advantageous to exclude rural regions from the search. This targeted approach prevents unnecessary examination of irrelevant areas, thereby optimising the search efficiency. A hierarchical representation of the environment, where images that present a similar appearance are grouped together in nodes, can significantly reduce the search space when finding similar places. As such, the hierarchy helps accelerate the retrieval process by skipping multiple nodes that are not relevant altogether.

2.4.2 Scalable Map Representations

Robot environmental maps captured using cameras are generally modelled using two representations: metric and topological. Metric representations, which define the geometry of the environment through quantitative dimensions such as distances and angles, are commonly employed in many robotic mapping applications. Thus, metric maps represent the world as accurately as possible, wherein the objects or keypoints are placed with precise coordinates. Many leading approaches rely on estimating correspondences between 2D keypoints in the query and 3D points in a sparse model using local descriptors. In this regime, [107] and [108] are considered state-of-the-art approaches in terms of accuracy when utilising handcrafted local features. A hierarchical localisation approach, HFNet [109], employs a monolithic CNN that simultaneously predicts local features and global descriptors for accurate 6-DoF localisation. By leveraging learned descriptors, strong localisation robustness across large variations of appearance was achieved. Metric maps are sensitive to noise, as they retain a

large amount of information about the environment, such as distances, measures, or sizes. Furthermore, these metric maps are more difficult to build and maintain and are computationally demanding. Consequently, localisation approaches that maintain a full metric map on a mobile device or robot are often restricted to small-scale environments due to the high memory requirements. As a result, while metric representations are valuable for their precision, their practical utility is bounded by constraints related to computational resources and memory capacity, especially in large-scale or long-term deployment scenarios.

On the other hand, topological maps model the environment using higher-order objects and their relationships using graphs, in which the nodes represent objects or places and edges correspond to the paths. These maps are simple and compact, scale better, and require significantly less storage than metric maps. Furthermore, they facilitate faster processing and efficient memory utilisation, making them ideal for extensive or long-term navigation tasks. There have also been a number of works suitable for very large-scale mapping and localisation without using an explicit metric representation [110, 39, 45]. To this end, [31] demonstrates a real-time, online, appearance-based topological SLAM algorithm that leverages the BoVW paradigm to represent the images and a discrete Bayes filter to compute the probability of loop closure. Several other works have been tailored to represent the environment discretely using occupancy grids, landmarks, and locations [111, 112, 113]. In a more recent work, Topomap [114] transforms a sparse feature-based map from a visual SLAM system into a three-dimensional topological map. [115] introduced a vision-based localisation approach that learns from the output of LiDAR-based localisation methods. In [116], the environment is represented with nodes with associated semantic features that are interconnected using coarse geometric information. We note that some of these methods are hybrid approaches that incorporate metric information on topological maps or vice versa to facilitate scalability. Focussed on using topological maps without the use of metric information, [117] demonstrated localisation using a two-level hierarchy for faster image retrieval in a topological map. Although such research has shown the potential for hierarchical matching, limited consideration has been given to the suitability and comparative performance of different feature representations used within these approaches. In our work, we emphasise utilising topological mapping, incorporating a hierarchical structure to expedite image retrieval through a selective search strategy.

In summary, in this thesis, we investigate the development of robust learned image

descriptors to enhance retrieval accuracy and efficiency, detailing the training regime that includes preparing subsets of data for stable contrastive training with a tailored loss function designed to yield optimal image representations. Furthermore, this research explores the novel unsupervised VAE-based compact image representations to facilitate fast and efficient scene categorisation, which are subsequently utilised for hierarchical image mapping and retrieval. We specifically target scalability and efficiency, aiming to effectively map and localise within trajectories spanning several kilometres. We seek to address the challenges of mapping extensive regions by leveraging a hierarchical framework incorporating the use of learned descriptors to facilitate rapid retrieval over large distances while maintaining localisation performance.

Chapter 3

VPR Systems: Datasets, Tools and Metrics

3.1 Introduction

Despite the significant advance in VPR research over the last two decades, state-of-the-art systems are still challenged by factors including strong fluctuations in illumination, changes in viewpoints, scene dynamics, variations due to weather and seasons, and the presence of occlusions. These factors can significantly alter the appearance of a place, posing a challenge to the system's ability to recognise it reliably.

Among the promising approaches to overcome these challenges are the development of advanced feature extraction algorithms, the integration of deep learning methodologies, and the curation of bespoke data subsets for neural network training and evaluation. A persistent obstacle in VPR research is the complexity of exploring, analysing, and determining the suitability of publicly available datasets, followed by the challenges of processing and managing them. Researchers often face difficulties due to inconsistent metadata, incompatible Software Development Kits (SDKs), and varying data formats and standards.

To address these issues, we introduce the first contribution of this thesis: a unified framework and software platform for processing and visualising VPR and odometry benchmark datasets. This framework, named OdoViz, streamlines the exploration,

analysis, and curation of data from a wide range of public datasets, providing tools for efficient sampling, visualisation, and standardisation. This facilitates the creation of tailored data subsets for training, validation, and testing, supporting the development of more robust and generalisable VPR systems. OdoViz has been published as *OdoViz: A 3D Odometry Visualisation and Processing Tool* [24] at the 2021 IEEE Intelligent Transportation Systems Conference (ITSC).

Additionally, we examine the critical aspects of evaluating and benchmarking VPR systems. We also explore the various datasets that are instrumental in training and testing these systems, examine the tools that facilitate their development and refinement, and discuss the metrics that are essential for assessing their performance. Through this exploration, we aim to provide a comprehensive overview of the current state of VPR datasets, processing and visualisation tools, and various evaluation methodologies.

3.2 Datasets

Although the research on autonomous vehicles dates back to the early 1980s [118, 119, 120], the past two decades have witnessed dramatic progress in the field. A core ingredient of this progress has been the use of data-driven and, in particular, deep learning techniques. In order for these approaches to be possible, multiple research groups and companies have led significant efforts to collect and release large-scale annotated datasets. These datasets facilitate the training of new models and approaches while also providing a means of tracking and benchmarking progress on various research challenges within the field.

The advent of large, annotated general datasets has been instrumental in driving advancements in computer vision tasks over the last decade. Datasets such as Middlebury [121] for stereo and optical flow evaluation, ImageNet [96] for image classification, SUN Database [122] for scene recognition, MS COCO [97] for image recognition, detection, and segmentation, PASCAL VOC [123] for object detection and segmentation, Places [124] for scene and object recognition, and ADE20K [125] for scene parsing and segmentation have become essential in the development, training, and benchmarking of various models aimed at solving computer vision tasks. In addition to offering a diverse array of annotated images for training data-driven models, these datasets also

serve as benchmarks for gauging progress and comparing methodologies across various computer vision tasks. These datasets have fundamentally transformed the landscape of computer vision, enabling significant progress in tasks such as image recognition, object detection, and face recognition. These datasets, pivotal in training data-driven models, often present images where a single object is the primary focus against a relatively uncomplicated background. This simplicity, while beneficial for specific computer vision tasks, starkly contrasts with the complexity encountered in images captured from vehicles, which are integral to driving datasets used for VPR.

Images derived from driving contexts are characterised by their dynamic and cluttered nature, incorporating multiple objects, vehicles, and pedestrians within complex scenes. Such images frequently exhibit greater levels of occlusion, where objects of interest are partially hidden by other elements in the environment, considerably elevating the complexity of the scene. This encompasses the differentiation between dynamic (movable) and static (non-movable) objects, including vehicles, pedestrians, roads, buildings, and other urban infrastructures. The accurate recognition of previously visited places, crucial for VPR systems, necessitates the interpretation of intricate scenes, understanding the underlying semantics, and discerning the spatial relationships and interactions among diverse elements within a scene.

Furthermore, the varying nature of outdoor environments, characterised by changing weather conditions, varying lighting, and seasonal transformations, presents unique challenges that general datasets may not fully encapsulate. Driving datasets, often captured using mobile robots or vehicles equipped with an array of sensors, are instrumental in addressing such requirements. In addition to offering video and image sequences, these datasets often include extensive annotations, including GPS coordinates, vehicle odometry, and environmental metadata.

Hence, given their importance within VPR research, in the following section we provide a comprehensive review of several of the more important automotive datasets, highlighting their distinctive characteristics, evaluating their significance, and discussing their impact as well as their individual constraints.

Figure 3.1: New College Dataset [110]

3.2.1 Core Research Datasets

In our research, we leverage these datasets in three primary ways: (1) directly, as a foundation for training our data-driven models and fine-tuning them to improve accuracy and reliability; (2) indirectly, by employing off-the-shelf models that have been pretrained on these datasets to further our objectives; and (3) for visualisation, comparative analysis, and preliminary validation and assessment.

3.2.1.1 New College

The New College dataset [110], released in 2008, serves as a foundational resource for research in robotics and autonomous navigation, focussing on the challenges of place recognition and mapping. Captured in and around the University of Oxford, two distinct subsets are presented: the New College and City Centre sequences. The New College sequence includes 2,146 images captured over a 1.9 km trajectory within the New College area, featuring several instances of loop closures. The images were obtained using a camera mounted on a pan-tilt mechanism attached to a robot, programmed to capture images to the left and right based on odometry every 1.5 metres; see Figure 3.1.

Cameras

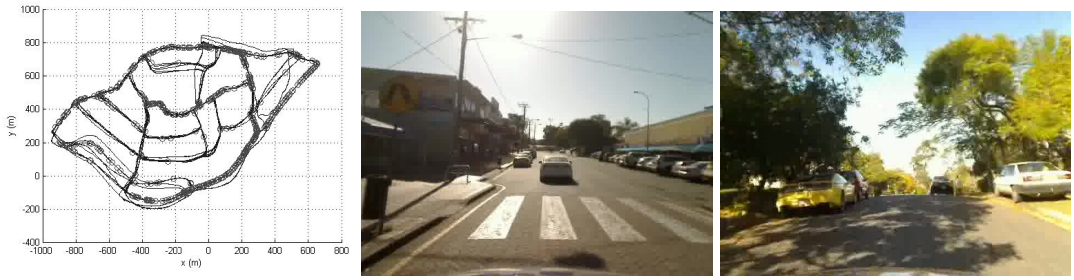


Figure 3.2: St Lucia dataset [46]

- Camera Setup: Single camera on pan-tilt
- Image Format: 640 x 480, JPG

The New College environment, characterised by visually repetitive structures such as a medieval cloister and a uniform garden wall, thereby testing the system's ability to handle perceptual aliasing. Conversely, the City Centre subset, encompassing a 2 km loop traversed twice in urban settings with moving elements like vehicles and pedestrians, calls for the handling of dynamic scenes. These subsets collectively serve as a testbed for advancing place recognition and mapping techniques in robotics, reflecting real-world navigation challenges.

3.2.1.2 St Lucia

The St Lucia [46] dataset is among the early public datasets to capture weeklong variations at multiple times of the day. Introduced in 2010, it encompasses 10 distinct sets of data collected over a suburb in St Lucia, Brisbane, driven through a network of streets. The datasets were captured under consistent sunny weather conditions over a span of two different periods, each spanning a few days; see Figure 3.2.

Cameras

- Monocular RGB Logitech QuickCam Pro 6000.
- Image Format: 640 x 480 pixels @15 Hz JPG.
- Mounted on the windscreen facing forward.
- FOV: 62° horizontal, 48° vertical.

Sequences

- First 5 sequences captured over 4 days in Aug 2009.
- Remaining 5 sequences captured over 2 days in Sep 2009.
- Duration: 20-25 mins.
- Start times: 8:45am, 10:00am, 12:10pm, 2:10pm, and 3:45pm.

An important aspect of the St Lucia dataset is its focus on a consumer-viable setup compared to other setups, which might use high-resolution, omni-directional cameras and custom controls. The dataset emphasises the use of commonly available equipment to gather data, making it a more accessible and broadly applicable resource for research and development in VPR and SLAM.

3.2.1.3 KITTI

The KITTI dataset [126] released by the Karlsruhe Institute of Technology (KIT) and Toyota Technological Institute (TTI) Chicago in 2012 serves as a prominent standard for evaluating autonomous vehicle systems and robotics. The KITTI Odometry dataset offers a comprehensive platform for the development and evaluation of algorithms for tasks such as odometry, localisation, and 3D mapping, which are crucial for autonomous navigation. The KITTI Odometry dataset is specifically tailored for assessing the performance of visual odometry algorithms and VPR systems. This dataset was collected in and around the city of Karlsruhe, Germany, using a standard station waggon fitted with a range of sensors; see Figure 3.3. The equipment consisted of high-resolution colour and greyscale stereo cameras, a Velodyne LiDAR scanner, and a GPS/IMU navigation system. The data was collected under various weather conditions and at different times of the day to ensure diversity and representativeness. The KITTI Odometry dataset comprises a series of sequences, each containing synchronised stereo images, LiDAR point clouds, and ground truth poses obtained from the GPS/IMU system. These sequences are divided into training and testing sets, with ground truth available only for the training set. Ground truth for the evaluation data is withheld for the purpose of providing public benchmarks.

Cameras:

Figure 3.3: KITTI dataset [126]

- Two high-resolution colour and greyscale stereo video cameras.
- Resolution: 1241 x 376 pixels @10 Hz.
- Mounted on the car's roof at approximately 1.5 metres above ground.

Sequences:

- 22 stereo sequences, with a total length of 39.2 km.
- Split into 11 sequences (00-10) with available ground truth for training.
- 11 sequences (11-21) reserved for testing purposes.

The limited representation of varied landscapes could potentially limit the generalisability of the results obtained from using this dataset. Additionally, the dataset does not include challenging weather conditions, such as heavy rain or snow, which are important factors to consider for autonomous navigation systems operating in different environments.

3.2.1.4 CMU Visual Localisation

The CMU (Carnegie Mellon University) Visual Localisation dataset [43] marks another advancement in the field of VPR and autonomous navigation, particularly in addressing the challenges posed by environmental changes across different seasons. Being made publicly available in 2012, it was developed as part of a broader effort to enhance real-time topometric localisation capabilities for autonomous vehicles and robotics,

Figure 3.4: CMU CVG dataset [43]

focussing on the robustness of these systems against the dynamic visual appearance of environments through the seasons. It includes a comprehensive collection of images captured across a diverse range of environmental conditions, including various weather scenarios, times of day, and, most notably, across different seasons. The traversals were on an 8.5-kilometre circuit, passing through both central and suburban Pittsburgh; see Figure 3.4. The trip was taken 16 times, with intervals ranging from 2 weeks to 2 months.

Cameras

- Two RGB cameras.
- Image Format: 1024 x 768 pixels, JPG.
- Mounted to the side of an SUV at 45°.

Data collection involved repeated traversals of the same urban and suburban routes around the Carnegie Mellon University (CMU) campus and the surrounding Pittsburgh area. The dataset includes images captured during the day and night in clear, cloudy, and rainy weather and across the distinct visual transformations presented by the four seasons: spring, summer, fall, and winter, captured over an extended period of 12 months. Consequently, it has become a pivotal benchmark for evaluating the ability of VPR and SLAM systems to withstand the visual challenges caused by seasonal changes, leading to significant advancements in algorithmic robustness and adaptability. The

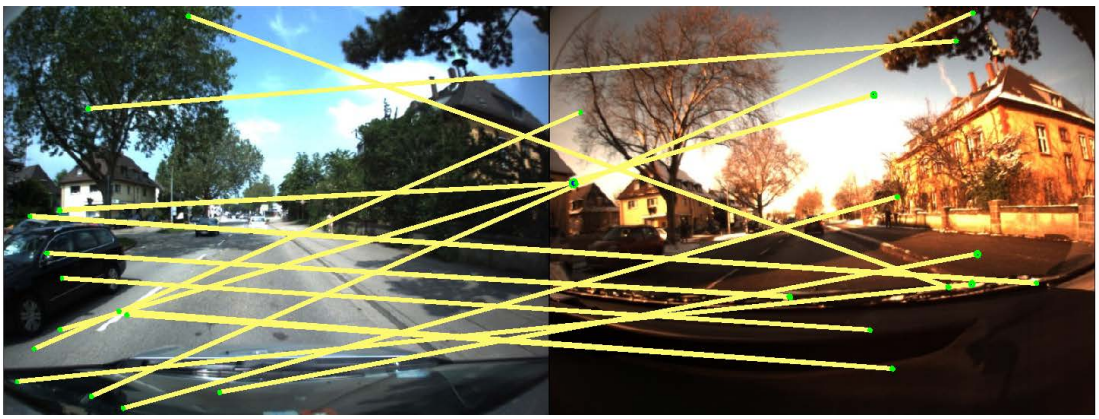


Figure 3.5: SURF features do not match reliably due to a substantial visual change in the Freiburg Across Seasons dataset [128].

dataset is no longer accessible[1] on the official site[2]. However, a curated subset of query and reference images was made available for benchmarking purposes[3] [127].

3.2.1.5 Freiburg Across Seasons

The Freiburg Across Seasons Dataset [128] seeks to address the challenge of visual localisation under extreme perceptual variations. Visual localisation in environments that undergo significant perceptual changes due to seasonal variations has been identified as a critical hurdle in the field of robotics and autonomous systems. The reliance on image matching through features like SURF [9] and SIFT [8], while effective under rotation and scale variations, has proven inadequate under conditions of extreme perceptual change, as shown in Figure 3.5. Recorded over three distinct periods — May 2012, Winter 2012, and May 2015 — the dataset comprises image sequences obtained from a car equipped with a forward-facing stereo camera; see Figure 3.6. The recordings span a cumulative distance of 70 km across Freiburg, Germany, capturing seasonal variations between summer and winter.

Cameras

- Forward-facing Bumblebee Stereo Camera
- Mounted outside in summer and inside in winter

[1] At least since 2018 and until the submission date of this thesis

[2] http://3dvis.ri.cmu.edu/data-sets/localization/

[3] https://www.visuallocalization.net/datasets/

Figure 3.6: Freiburg Across Seasons dataset [128]

- Image Format: 1024 x 768 pixels, JPG

Sequences

- May 2012: 6,915 images @1 Hz, 10 km trajectory
- Winter 2012: 30,790 images @4 Hz, 50 km trajectory
- May 2015: 5,392 images @4 Hz, 10 km trajectory

3.2.1.6 Pittsburgh

The Pittsburgh dataset [129, 130], often called the Pitts250k dataset, sources its images from Google Street View captures of Pittsburgh, employing a method to crop equirectangular panoramas into tiles followed by gnomonic projections to create perspective images; see Figure 3.7. Subsequently, the dataset compiles 254,064 perspective images derived from 10,586 Street View panoramas. Each panorama, measuring 6,656 by 3,328 pixels, is processed to generate 24 distinct perspective images. The dataset captures database and query images with a two-year interval; however, there are no noticeable weather variations.

Image Specifications

- Image Format: 640 x 480 pixels, JPG
- Field of View: 60° HFoV across 2 pitch and 12 yaw directions

The ground truth relies on GPS data of Street View panoramas, which often generalise locations to the street's median, introducing a positional accuracy ranging between

Figure 3.7: Pittsburgh Dataset [130]

7 and 15 metres. The test set, containing 24000 perspective images, was generated from 1000 panoramas randomly selected from 8,999 panoramas of the Google Pittsburgh Research Data Set. Encompassing different sessions, varying viewpoints, and illumination changes, the dataset set a high benchmark for evaluating place recognition systems. Pitts30k, a subset of the larger Pitts250k dataset, facilitates the examination of various VPR algorithms under a more constrained dataset size while maintaining the diversity and challenge presented by the broader dataset.

3.2.1.7 Tokyo 24/7

The Tokyo 24/7 dataset [131] presents a relatively large database also derived from Google Street View imagery. Published in 2015, the database of geo-tagged images includes 75984 views generated from the original 6332 street-view panoramas and 597744 synthesised views generated at 49812 virtual camera positions; see Figure 3.8. The test set includes 1125 query images captured using Apple iPhone 5s and Sony Xperia smartphones, captured at 125 distinct locations, each at 3 different viewing directions and at 3 different times of day.

Image Specifications

- Image Format: 1280 x 960, JPG
- Field of View: 60° HFoV

Figure 3.8: Tokyo 24/7 Dataset [131]

The dataset was created to evaluate the robustness of image retrieval algorithms under challenging conditions, such as extreme changes in illumination and viewpoint. The Tokyo 24/7 dataset, along with the Pittsburgh dataset, is commonly used in pretraining of NetVLAD [67], a popular CNN for contrastive learning.

3.2.1.8 Cityscapes

Cityscapes Dataset [132], which was made publicly available in 2015, provides a comprehensive suite of high-quality annotated images captured in diverse urban settings across several European cities. Designed to advance the development of pixel-level and instance-level semantic labelling, the dataset facilitates the training and evaluation of semantic segmentation models in interpreting complex urban landscapes; see Figure 3.9. Cityscapes focuses on urban settings, with annotations from 50 different cities and towns in and around Germany, captured over several months, showcasing the diverse effects of different seasons, specifically spring, summer, and autumn. It includes over 5,000 finely annotated images and an additional 20,000 images with coarse annotations. The dataset is structured to support semantic segmentation of 30 different classes related to urban navigation, such as roads, vehicles, and pedestrians. It was originally recorded as video; however, the frames were manually selected to have a large number of dynamic objects, varying scene layouts, and varying backgrounds.

Image Specifications

Figure 3.9: Cityscapes Dataset [132]

- Image Format: 2048 x 1024 pixels @17 Hz, PNG
- 16-bit HDR, debayered and rectified
- 8-bit RGB LDR also provided.
- Sampled individual non-sequential images.
- Fine annotations: 5000 images from 27 cities, manually selected.
- Coarse annotations: 20000 images from 23 cities, every 20 m or 20 s, whichever is earlier.
- Vehicle odometry, GPS information, and outside temperature are available.

Furthermore, the Cityscapes dataset, while extensive, does not originally encompass conditions such as fog, rain, or snow, potentially limiting its applicability in adverse weather scenarios. To address these gaps, subsequent contributions from the research community have augmented the dataset with additional annotations and modifications. In 2019, the Cityscapes dataset was enriched with the introduction of panoptic labels,

a significant advancement designed to unify the tasks of semantic segmentation and instance segmentation into a cohesive framework. In 2020, Cityscapes 3D [133], was announced as an extension of the original Cityscapes with 3D bounding box annotations for all types of vehicles as well as a benchmark for the 3D detection task. At the time of writing, the Cityscapes dataset provided a comprehensive benchmark suite and an evaluation server, facilitating the assessment of models across several key tasks:

- Pixel-Level Semantic Labelling: This involves assigning a class label to each pixel in an image, enabling a detailed understanding of the scene.
- Instance-Level Semantic Labelling: Beyond classifying pixels, this task distinguishes between different instances of the same class, such as individual vehicles or pedestrians.
- Panoptic Semantic Labelling: Panoptic labelling merges the tasks of semantic and instance-level semantic labelling to provide a unified scene understanding.

Given its scale, diversity, and dense pixel-level annotations, the dataset has emerged as a seminal resource in autonomous driving and urban scene understanding within computer vision research. Consequently, the dataset has become a prominent resource for the pretraining of data-driven models aimed at semantic segmentation and instance segmentation in automotive settings. This utility underscores its value in the foundational stages of model development, where a broad and generalised understanding of driving scenes is crucial. Nevertheless, a notable limitation of the dataset is its exclusive focus on images captured during daylight hours. This absence of low-light conditions, such as those encountered at dusk, dawn, or night, presents a challenge in training models to generalise effectively across a wider range of lighting conditions.

3.2.1.9 Oxford RobotCar

The Oxford RobotCar dataset [7] provides an autonomous driving dataset focussed on long-term autonomy in changing urban environments. Released in 2017, this dataset is notable for its extensive collection of images and sensor data, recorded over a year in Oxford, UK, encompassing a wide range of weather conditions, traffic scenarios, and times of day; see Figure 3.10. It was collected using an autonomous Nissan LEAF car

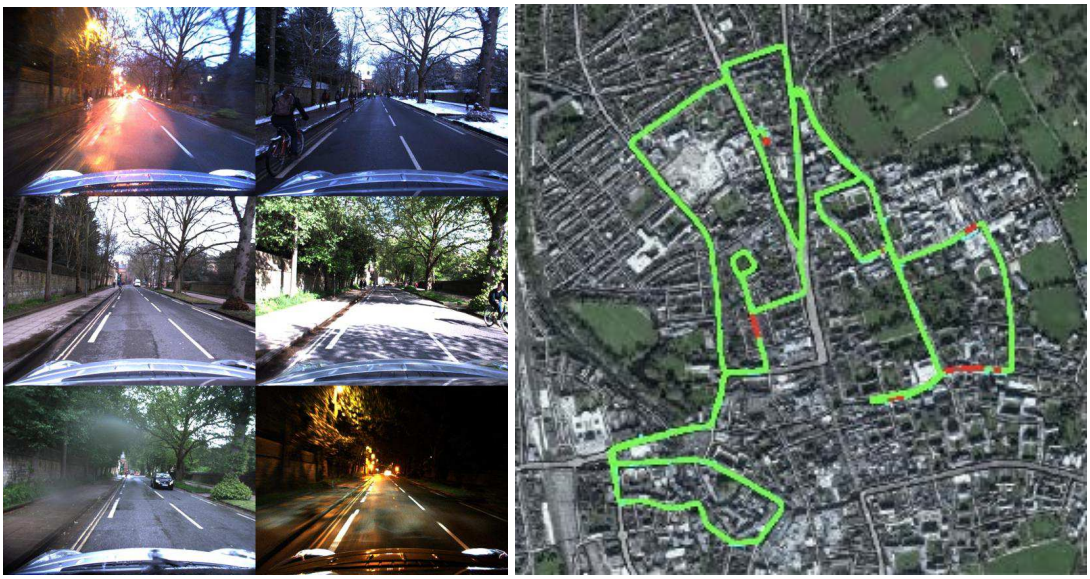


Figure 3.10: Oxford RobotCar Dataset [7]

equipped with a suite of sensors, including cameras, LIDARs, GPS, and INS. The data was recorded over repeated traversals of a consistent urban route in Oxford, covering approximately 10 km. Over 100 repetitions of this route allowed for the capturing of the same scenes under different conditions, providing a unique opportunity to study the effects of long-term environmental changes on autonomous driving systems.

Cameras

- Point Grey Bumblebee XB3 trinocular stereo and 3x Grasshopper2 monocular
- Stereo HFoV 66°: 1280 x 960 pixels @16 Hz, Bayer GBRG PNG
- Monocular HFoV 180°: 1024 x 1024 pixels @11.1 Hz, Bayer RGGB PNG

Sequences

- 2 different routes
- Over 100 sequences

The dataset's diversity and scale set it apart from other datasets available at the time. The car traversed approximately twice a week on average over the period of May 2014 to December 2015, collecting almost 20 million images from 6 cameras mounted to

the vehicle. The dataset covers a wide range of environmental conditions, including different times of day (from dawn till dusk), varying weather conditions (sunny, rainy, cloudy, and overcast days), and different seasons, providing a comprehensive view of how urban landscapes change over time. Consequently, the dataset has significantly impacted various aspects of autonomous vehicle research and robotics. Its comprehensive coverage under varying conditions has been crucial in developing algorithms that are robust against environmental changes, an essential attribute for reliable autonomous navigation. The longitudinal aspect of the dataset offers rare insights into the evolution of urban landscapes, aiding in the development of adaptive systems that are key for long-term autonomy. Along with the CMU Seasons [43] dataset, the Oxford RobotCar dataset has established itself as a benchmark in the field, enabling researchers to evaluate and compare the performance of SLAM, VPR, and other autonomous driving algorithms in a consistent yet challenging environment [127]. Furthermore, the vast and varied data collection has been instrumental in advancing deep learning models, especially in object detection, scene segmentation, and environmental understanding.

3.2.1.10 BDD100K

Introduced as part of the Berkeley DeepDrive project, the BDD100K [134] dataset is one of the largest and most diverse of its kind, featuring a vast collection of video clips and images that reflect the complexities and variabilities inherent in driving scenarios across different geographical, environmental, and urban contexts. It comprises 100,000 video clips, each 40 seconds long, collected from over 50,000 rides across the United States, covering New York, Berkeley, San Francisco, and the other regions in the Bay Area. These clips are accompanied by frame-level annotations, including labels for objects, lanes, drivable areas, and full-frame instance segmentation; see Figure 3.11. The dataset covers various weather conditions, times of day, and urban and rural scenes in its 120 million images, making it an extensive resource for training and evaluating computer vision models under realistic driving conditions.

Sequences

- 100,000 sequences, each 40 seconds long
- Image format: 1280 x 720 pixels @30 Hz

Figure 3.11: BDD100K dataset [134]

The diversity offered by the dataset is crucial for developing robust computer vision algorithms that can adapt to varied real-world conditions, including different geographical locations, weather, and lighting. Furthermore, the dataset's extensive and detailed annotations for a range of objects and scene elements are instrumental in advancing object detection and scene understanding. These comprehensive annotations enable the creation of more accurate and reliable models for complex driving environments.

3.2.2 Auxiliary Datasets

Certain datasets were not utilised in our research, either because they did not align closely with our specific research goals or because they were introduced during or after the timeframe of our individual studies. Nevertheless, it is important to acknowledge these recent contributions to the field, as they represent significant advancements and offer valuable resources for future work in VPR and related areas of study. For the sake of completeness, we also provide an overview of these datasets, detailing their unique features and potential applications, to acknowledge their role in advancing the scope of current and future investigations in the area.

3.2.2.1 San Francisco

The San Francisco dataset [135], released in 2011, provides a large database collected by a car-mounted camera. The dataset contains approximately 150k panoramic images captured at 4-metre intervals that are then converted to approximately 1.7 million perspective images. Aimed at city-scale landmark recognition from mobile devices, the dataset includes 803 cell phone query images tagged with a mix of real and simulated GPS coordinates. These query images include environmental and urban clutter, varying

light conditions, reflections, and significant perspective shifts that introduce photometric and geometric distortions between the query images and their corresponding entries in the database.

3.2.2.2 UQ St Lucia

The UQ St Lucia dataset [136] was tailored for stereovision-based SLAM. It was captured from a car driven along a 9.5-kilometre circuit around the University of Queensland's St Lucia campus, capturing a wide array of urban scenarios. Captured in December 2010, the dataset encapsulates diverse environmental conditions, including roadworks, speed bumps, varying illumination levels, and complex traffic situations such as multi-lane roads and roundabouts.

The ground truth GPS data is provided using XSens MTi-g INS/GPS at 120 Hz, along with a USB NMEA GPS for additional localisation data at 1 Hz. The stereo camera setup underwent a calibration procedure using over 150 checkerboard image pairs. This setup renders the dataset suitable for a wide range of computer vision tasks, including but not limited to stereo depth estimation, 3D reconstruction, and visual odometry.

3.2.2.3 Mapillary Vistas

The Mapillary Vistas dataset [137], introduced in 2017, is a large-scale street-level image dataset containing 25,000 high-resolution images annotated into 66 object categories, of which 37 classes are instance-specific labels. This was later augmented in v2.0 to cover 124 object categories, 70 of which bore instance-level labels. The dataset provides dense and fine-grained semantic annotations by using polygons for delineating individual objects.

3.2.2.4 Visual Localisation Benchmark

Visual Localisation benchmarking platform[4] [127] introduced a benchmark for estimating the 6 Degrees of Freedom (DoF) camera pose relative to a reference scene,

[4]available at https://www.visuallocalization.net

emphasising the robustness of localisation methods under diverse environmental conditions. While 6-DoF estimation plays a crucial role in enhancing experiences in virtual reality (VR) and augmented reality (AR), its significance extends to the robotics domain, particularly as a key technology in autonomous navigation in self-driving vehicles and mobile robots. Published in 2018, the platform's significance lies in its development of the first benchmark datasets specifically designed to evaluate the impact of varying conditions, such as day-night changes, weather, and seasonal variations, on the accuracy of 6 DoF camera pose estimations. The benchmark comprises three distinct datasets: Aachen, RobotCar Seasons, and CMU Seasons, which are derived from the Aachen Day Night dataset [138], Oxford RobotCar dataset [7], and CMU Visual Localisation dataset [43], respectively. Each dataset consists of a set of reference images with ground truth poses and a set of query images. A triangulated 3D model is also provided for each dataset, which can be used by structure-based localisation approaches. The benchmark platform adopts a unified evaluation protocol, considering both position and orientation accuracy, to ensure the comparability of results across different localisation methods and datasets, allowing for a comprehensive assessment of a method's robustness to the challenges posed by environmental changes over time. Datasets based on the SILDa Weather and Time of Day dataset [139], and the Symphony Seasons dataset [140] were later added. Furthering the expansion efforts, an extended version of the CMU Seasons dataset and an updated RobotCar Seasons dataset, with more trajectories and annotations, were introduced in 2020 [141]. Moreover, the platform extended its utility to indoor settings by incorporating datasets like InLoc [142], ETH MS [143], and the Gangnam Station and Hyundai Department Store from Naverlabs [144]. These inclusions have methodically augmented the dataset repository, facilitating comprehensive evaluations across a spectrum of visual localisation challenges.

3.2.2.5 Cross Seasons

The Cross-Season Correspondence Dataset for Robust Semantic Segmentation [145], introduced in 2019, aimed at enhancing the robustness of semantic segmentation models under diverse environmental conditions. By utilising 2D-2D point matches between images captured across different seasons, weather conditions, and times of day, the dataset enables the training of convolutional neural networks (CNNs) that maintain labelling consistency despite the changing environmental conditions. The use of geo-

metric 3D consistency for establishing point correspondences bypasses the limitations of photometric information, which can vary significantly with lighting and atmospheric conditions, thus involving minimal human intervention, relying instead on the geometric consistency between point clouds. It comprises two distinct subsets, derived from the Extended CMU-Seasons dataset [127] and the Oxford RobotCar Seasons dataset [127], respectively. The CMU Seasons Correspondence Dataset contains 28766 image pairs from different seasonal conditions, while the Oxford RobotCar Correspondence Dataset contains 6511 image pairs covering different seasonal and illumination conditions.

3.2.2.6 WoodScape

The WoodScape dataset [146], a unique collection tailored for autonomous driving research published in 2019, particularly emphasises fisheye camera imagery, diverging from the standard rectilinear perspective datasets commonly found in the field. The omnidirectional field of view offers a comprehensive perspective on the environment, benefiting tasks that require broader spatial understanding and contextual awareness. Fisheye cameras are increasingly used in commercial automotive systems for their wide-angle field of view to reduce blind spots and enhance overall coverage of the vehicle's surroundings, thereby reducing the need for multiple cameras and lowering overall system costs. The use of fisheye images challenges traditional VPR algorithms to adapt and extract relevant features from distorted images, pushing the development of more versatile and robust solutions. WoodScape comprises four surround-view fisheye cameras with a 190° HFoV facing front, rear, left, and right, and nine tasks, including segmentation, depth estimation, 3D bounding box detection, and soiling detection. Semantic annotation of 40 classes at the instance level is provided for over 10,000 images. The dataset advocates solutions that can work directly on raw fisheye images, modelling the underlying distortion. Designed to complement existing automotive datasets with limited FoV images, the dataset encourages the implementation of multi-task networks that consume annotations for multiple tasks concurrently. The focus on fisheye cameras, however, limits the dataset's applicability for systems designed around narrow-field camera frameworks that remain prevalent in the automotive industry for high-speed, front-facing driving scenarios.

3.2.2.7 nuScenes

The nuScenes dataset [147], introduced by Aptiv in 2020, is a comprehensive data collection designed for autonomous driving research. Notably, this dataset is among the first to provide a full autonomous vehicle sensor suite, including 6 cameras, 5 radars, 1 LIDAR, GPS, and IMU, covering a rich variety of urban landscapes, captured in diverse lighting and weather conditions across Boston and Singapore. The dataset includes over 1,000 driving scenes, amounting to 1.4 million camera images, along with corresponding LIDAR and radar data and 3D bounding box annotations for various object categories. The dataset includes nighttime scenes, adverse weather conditions, and a multitude of dynamic objects, highlighting its potential in VPR research.

3.2.2.8 Mapillary Street Level Sequences

The Mapillary Street-Level Sequences (MSLS) dataset [148] is a large-scale, diverse dataset designed for the task of lifelong place recognition. Released in 2020, the dataset comprises over 1.6 million images curated from the Mapillary collaborative mapping platform. The dataset is structured into a large number of short sequences of street-level images, covering urban and suburban settings across 30 cities across six continents. The dataset includes images tagged with sequence information and geo-located with GPS and heading angles, captured across various times of the day and year, spanning all seasons over a nine-year period.

3.2.2.9 San Francisco XL

The San Francisco eXtra Large (SF-XL) dataset [77], released in 2022, was constructed from Google StreetView imagery. The dataset offers an expansive, densely covered, and temporally varied compilation of data. SF-XL combines 3.43 million equirectangular panoramas, subsequently segmented into 12 horizontal crops, yielding a total of 41.2 million images, each annotated with six degrees of freedom (6 DoF) information, encompassing GPS and heading data. The images in the dataset, taken between 2009 and 2021, encompass a significant temporal variation, thereby incorporating long-term environmental changes that add value to the dataset.

3.2.2.10 KITTI 360

The KITTI 360 dataset [149] extends beyond its predecessor, the KITTI dataset [126], by offering enhanced semantic and instance annotations in both 2D and 3D perspectives, alongside richer 360-degree sensory data through fisheye images and laser scans. Made available in 2022, it encompasses over 320,000 images and 100,000 laser scans collected across a driving distance of 73.7 kilometres in the suburbs of Karlsruhe, Germany. The dataset includes accurate geo-localisation of all frames, leveraging OpenStreetMap for precise positioning and the assignment of consistent instance IDs across frames for robust tracking and analysis. Semantic label definitions align with the Cityscapes dataset, facilitating cross-dataset comparisons and evaluations with 19 classes designated for assessment. The sensor setup for data acquisition includes a 180° fisheye camera on the left and right of the vehicle, a 90° perspective stereo camera at the front with a 60 cm baseline, and a combination of a Velodyne HDL-64E and a SICK LMS 200 laser scanning unit mounted on the roof, supplemented by an IMU/GPS localisation system. Offering a comprehensive 360° field of view, the dataset supports a wide range of tasks, including semantic segmentation, instance segmentation, semantic scene completion, and urban scene understanding.

3.2.3 Synthetic Datasets

In the development of VPR systems, the generation and application of synthetic datasets have been identified as a pragmatic approach to circumvent the limitations encountered in the collection and annotation of real-world data. Many real-world datasets, including the ones discussed in the previous subsections, face challenges in covering an exhaustive range of scenarios, objects, and environmental conditions due to the resource-intensive nature of their collection and the manual labour required for data annotation. Many of these datasets, while extensive, lack representation of rare or edge cases, which are critical for robust model training.

Synthetic datasets, generated through simulations, offer a broader array of scenarios without the need for manual annotation, addressing some of the key challenges in real-world data collection and preparation. Thus, they play a vital role in the field of autonomous driving, providing a means to train and evaluate algorithms under controlled

conditions that might be difficult, expensive, or even impossible to replicate in the real world.

Nonetheless, the capacity to generate environments that closely mimic real-world conditions, including the changes that occur across different seasons, is crucial for VPR datasets. As such, without the ability to accurately replicate these conditions, the potential for synthetic datasets to contribute meaningful insights or improvements to VPR systems is severely compromised. Furthermore, the use of synthetic datasets introduces the challenge of the domain gap, which is the difference in data characteristics between synthetic and real-world environments. This gap can affect the performance of models trained on synthetic data when applied to real-world tasks, as the variability and complexity of natural environments are not fully replicated in synthetic scenarios. During the period in which this research was conducted, the state of generative imaging techniques and rendering methods did not reach a level of advancement necessary to justify the utilisation of synthetic datasets for tasks associated with VPR. Therefore, our research methodology was tailored to rely on real-world datasets, which, despite their own set of limitations, offered a more reliable basis for training data-driven VPR techniques.

In this section, we will discuss a range of synthetic datasets that, while not currently in use for our specific research, stand out for their significance in the broader context of computer vision and autonomous driving studies.

3.2.3.1 Synthia

Synthia [150] is a seminal synthetic dataset designed for semantic segmentation and urban scene understanding. It showcases the early use of synthetic data to supplement real-world datasets, emphasising the generation of urban landscapes under various conditions. The dataset has been further enhanced by the introduction of Synthia-SF [151] and Synthia-AL [152] in 2017 and 2019, respectively, each addressing unique facets of urban environment simulation. Synthia-SF [151], introduces a novel depth model based on Slanted Stixels. This model offers an improved representation of non-flat roads, a common challenge in urban scene analysis, although with a trade-off in computational efficiency. Synthia-AL [152], focussing on active learning, includes a comprehensive set of classes such as void, sky, building, and traffic elements. It extends

its utility by offering detailed ground truth data, encompassing instance segmentation, 2D and 3D bounding boxes, and depth information.

3.2.3.2 Virtual KITTI

Inspired by the real-world KITTI [126] dataset, Virtual KITTI [153], replicates the original's driving scenarios in synthetic form. Announced in 2016, it aims to provide a controlled environment for testing computer vision algorithms, including those for VPR. By mimicking real-world conditions in a virtual setting, Virtual KITTI allows for the exploration of various scenarios and conditions, albeit within the constraints of its synthetic nature. An enhancement of this dataset, Virtual KITTI 2 [154], proposed in 2020, provides a more photo-realistic and feature-rich dataset, exploiting improvements of the Unity game engine to provide new data such as stereo images and scene flow.

3.2.3.3 CARLA

The CARLA (Car Learning to Act) [155] simulator has been instrumental in generating synthetic datasets for autonomous driving research, including VPR. However, datasets produced within CARLA often face criticism for lacking sufficient photorealism, which can impede their effectiveness in scenarios requiring high fidelity to real-world appearances. This limitation highlights the challenge of using CARLA-generated datasets for direct application in tasks demanding accurate representation of real-world conditions.

3.2.3.4 Synscapes

Recognised for its advanced approach to synthetic dataset generation, Synscapes [156] employs procedurally generated scenarios and photorealistic rendering to closely match the real-world Cityscapes [132] dataset. This methodological sophistication has shown to enhance transfer learning capabilities significantly, setting a benchmark for photorealism and domain specificity in synthetic datasets for driving scenarios.

3.2.3.5 SynWoodScape

As a synthetic counterpart to the WoodScape dataset [146], SynWoodScape [157], released in 2022, attempts to replicate the latter's diverse driving scenarios in a virtual environment. Despite this ambition, it shares a common limitation with other CARLA-generated datasets in terms of photorealism. The lack of sufficient realism has been identified as a barrier to achieving improved model performance in the target domain, underscoring the necessity of domain adaptation when integrating synthetic and real-world data.

3.2.3.6 PD-WoodScape

Developed by Parallel Domain in 2023, PD-WoodScape [158] aims to bridge the gap between synthetic and real datasets by closely matching the WoodScape dataset's sensors, annotations, and operational domains. Rendered with a high-grade synthetic data pipeline, it achieves superior photorealism compared to CARLA [155] generated datasets. Special attention to annotation alignment ensures that training on PD-WoodScape avoids the pitfalls of false positives that may arise from discrepancies in dataset labelling, thereby minimising the synthetic-to-real domain gap.

3.3 Tools

The analysis and processing of public datasets for VPR research requires a combination of specialised tools and methodologies. Processing of autonomous driving datasets typically involves either directly accessing the data through custom code, utilising bespoke SDKs designed for individual datasets, using related tools from fields such as photogrammetry, or employing dedicated autonomous vehicle dataset frameworks.

3.3.1 Visualisation Tools

Visualisation tools are integral to VPR systems as they enable interpretation and analysis of the complex processes underlying place recognition tasks. These tools facilitate the

inspection of poses, images, and relevant metadata, matching mechanisms, and the spatial-temporal dynamics of poses, thereby providing insights into the strengths and weaknesses of the data required for learned VPR algorithms. By visually inspecting the correspondences between query and reference images, researchers can identify patterns and anomalies in the data associations made by the VPR system, leading to more informed decisions about algorithmic adjustments and enhancements. Thus, visualisation tools serve as a critical component in the development, evaluation, and optimisation of VPR systems.

Probably the most notable tool from the field of photogrammetry for visualisation of driving datasets is Visual SFM [159]. This tool allows loading images, performing Structure from Motion (SfM), saving and loading *.nvm* files, and viewing the resultant point cloud and image associated with each camera pose. Being a full-fledged SfM tool, its support does not extend beyond *.nvm* files and therefore does not support many of the additional file formats included with vehicle datasets.

More recently, a number of automotive companies have released tools designed specifically to address some of these issues and to better support research in the space.

Webviz [160] is a web-based tool, developed by Cruise, for general robotics data inspection. The tool provides visual insights on ROS bag files and allows connecting to a live robot or simulation. Webviz allows custom data visualisation layouts from a collection of configurable panels for displaying information like text logs, 2D charts, and 3D depictions of the vehicle's environment. Facilitating playback and visualisation of ROS bag files, Webviz provides an intuitive platform for developers and researchers to dissect and understand the behaviour of robotic systems through visual data inspection.

Autonomous Visualisation System (AVS) [161] is an open and modular 3D visualisation toolkit developed by Uber, allowing visualisation of data across the autonomous vehicle development spectrum, focussing on perception, motion, and planning data. It uses XVis, which serves as an underlying data protocol, outlining a structured, stream-oriented approach to scene representation over time. It focuses on rendering performance and composability, utilising a React and WebGL-based visualisation platform to ensure real-time playback and smooth interaction with complex scenes. Offering a scalable and flexible solution for the visualisation and analysis of autonomous vehicle data, the tool allows loading and navigating through individual poses of a trajectory

while also featuring inspection of GPS/INS, point clouds, and other metadata on a frame-by-frame basis.

Although both Webviz and AVS provide strong frameworks for processing and visualisation, both are targeted at real-time visualisation and playback of vehicle sensor data, i.e., providing a replay functionality at the local level of the vehicle. As such, they are not directly applicable to the use cases considered here, where we wish to visualise and process complete trajectories at a global level.

Another visualisation tool of significance is FiftyOne [162], an open-source tool developed by Voxel51. It is designed to facilitate the analysis, visualisation, and management of large-scale image and video datasets used in deep learning projects. The tool facilitates the viewing and editing of annotations (e.g., bounding boxes, semantic segmentation) that describe the content of images. While Voxel51 excels in video and image analytics, VPR-specific visualisation requires integration with additional tools or libraries that are tailored to the spatial and geometric analyses fundamental to VPR.

We also note additions to the popular Open3D library [163], in particular the Open3D-ML extension, which extends the library to support common deep learning tasks in autonomous vehicle research. These extensions are targeted at 3D deep learning tasks such as 3D object detection and semantic segmentation.

3.3.2 Limitations

Despite the presence of the aforementioned tools, the process of managing and utilising public datasets remains complex and requires careful considerations. This involves several critical stages, each demanding a unique set of approaches and methodologies.

Typically, the first step in determining a dataset's applicability for a given project is to visualise and analyse its included trajectories. For example, assessing a dataset's suitability for VPR research requires analysing the geographical extent and degree of overlap between individual trajectories. Many popular public driving datasets only provide a single view of the included trajectories overlaid on a static aerial or satellite image (see Figure 3.12). More recently, datasets such as BDD100K have included executable scripts to visualise the top view of the individual trajectories. However, such scripts typically do not include functionality for loading multiple trajectories, analysing

(a) KITTI (b) New College (c) RobotCar (d) UQ St Lucia

Figure 3.12: Path overlaid on satellite images provided by different popular datasets

Dataset	Pose File Format	Camera Images
New College [110]	TXT	JPG
KITTI Odometry [126]	TXT	PNG
CMU Seasons [43]	NVM	JPG
UQ St Lucia [136]	LOG	Bayer PNG
Oxford RobotCar [7]	CSV	Bayer PNG
BDD100K [134]	JSON	MOV Video

Table 3.1: Different file formats used by popular driving datasets

the overlaps, and inspecting individual pose data (e.g., GPS, heading, image, and other sensor data).

Given a suitable dataset, utilising it requires the researcher to become familiar with its organisation and structure so as to curate suitable data for training and evaluating new models. Many file formats have been adopted by different datasets (as shown in Table 3.1), with each dataset having its own sensor types, positioning, and configuration. Ideally, software development kits (SDKs) are released alongside the datasets to simplify their use (e.g., PyKITTI [164], RobotCar SDK [165], Cityscapes Scripts [166], BDD100K Toolkit [167], etc.). However, in our experience, there is still a steep learning curve associated with most driving datasets. In particular, one has to expend considerable time and effort to set up and become familiar with the SDK. Furthermore, given the variations between the APIs, the use of different programming languages, etc., any code developed to perform additional tasks will have limited portability across datasets.

Supervised learners learn a function using the labelled training data, and hence, the quality and precision of such data need to be high [168, 169, 170, 171]. Creating training sets from one or more datasets usually involves (i) sampling or selecting the images

required, (ii) collecting corresponding information regarding the vehicle's pose from the Global Positioning System (GPS) and Inertial Navigation System (INS) data, (iii) applying positional and rotational offsets, and (iv) finding pose correspondences within or across the trajectories. For many computer vision challenges, it is often preferable to select images of importance based on their location on the map, e.g., choosing images of landmarks, images captured at intersections, or images of the same place captured from different viewpoints. To train computer vision models to extract features from images robust to different viewpoints, seasons, and different times of the day, thousands of image correspondences based on the GPS data are required. The SDKs are often limited to only parsing the pose files to obtain GPS/INS data for images and converting the images to standard file formats and do not include an interactive interface to perform subsampling or finding such correspondences.

3.4 OdoViz

To address the aforementioned problems, we present OdoViz, a novel extensible platform for 3D odometry visualisation and processing, providing a unified software framework for working with a wide array of heterogeneous odometry benchmark datasets. OdoViz is web-based, flexible, extensible, easy-to-use, and supports common odometry file formats with customisable scene and offset settings. The system allows the user to perform operations such as sampling, identifying, and comparing pose correspondences within and across multiple trajectories. It also allows loading, inspecting, visualising, and processing GPS/INS poses, point clouds, and camera images. OdoViz has built-in support for popular driving datasets, including: Oxford RobotCar [7], CMU Seasons [43], BDD100K [134], UQ St Lucia [136], New College [110], and KITTI [126], and supports user-defined extensions to support custom datasets. The system also includes plugins for importing and exporting settings, as well as extensions for a range of tasks, including (i) analysing top-k matches in an image retrieval benchmark of a feature extractor and (ii) visualising topological nodes along a loaded trajectory. Additional features can be implemented through custom extensions and plugins.

We explain the design and architecture of OdoViz in Section 3.4.1, discuss each of the core modules in Section 3.4.2, elaborate on sampling and finding correspondences functionalities in Section 3.4.3, and discuss the extensions and plugins in Section 3.4.4.

Existing VPR tools like Webviz and AVS address important needs within the community by providing rich frameworks for processing and visualisation at the local level of the vehicle, i.e., targeting egocentric tasks such as real-time visualisation and playback of vehicle sensor data and 3D object detection. However, they are not directly applicable in more global-level tasks such as the use cases considered in this section, e.g., identifying corresponding poses within or across trajectories, or for visualising loop closures.

To our knowledge, OdoViz is the only tool that supports loading, viewing, and processing of complete trajectories and performing common odometry tasks such as sampling and finding pose correspondences. OdoViz was initially created as a small in-house tool to load and inspect different datasets and to reduce the effort required to incorporate new datasets within our research. The tool has been in development since 2019, with multiple features added to the software since then to aid our research. It had grown mature to act as a generic, extensible 3D odometry visualisation and dataset curation tool, and we have open-sourced[5] the work under the MIT licence for the benefit of the wider research community.

A live instance[6] of OdoViz is hosted online for preview purposes. Documentation and a number of video tutorials[7] have been made available to assist in using the system and completing common tasks. Documentation on extending the system to support a new dataset is also provided.

3.4.1 Design

The OdoViz architecture consists of (i) a front end providing a *rich client* built on React, Redux, and Redux Saga, and (ii) a backend server designed to act as a JSON API-based *thin server* primarily for serving files. The reactive front end, equipped with a *Three.js* environment, provides a 3D user interface. The complete application is loaded as a Single Page Application (SPA), ensuring full functionality and minimum processing latency after the application is loaded into memory. Network connectivity is required only to load new files from the server.

[5] Source-code is available at https://github.com/robotvisionmu/odoviz

[6] Live instance is available at https://odoviz.cs.nuim.ie

[7] Video tutorials are available at https://www.youtube.com/playlist?list=PLKIavzsN4tuGi1SKDSPss0M8v4zswVEn9

The system is designed around the following principles:

- **Web-based:** The software is web-based and therefore runs without compilation on all major operating systems directly in the browser. It allows for the easy addition of new features and debugging of existing ones, with support for hot and live reloading. Dependency resolution is performed automatically using NPM (Node Package Manager), where *Webpack* links the dependencies, minifies, and bundles the app in a JavaScript file, which can then be served along with a static HTML file. Web-based tools further allow providing a rich, modern, reactive and easily modifiable user interface while being highly accessible and portable.

- **Extensible:** The system is designed with extensibility in mind. This is achieved by associating a single parser file with each dataset that includes paths to its various file locations, along with information used for extracting data from the info file and linking the data with the images. Hence, adding support for a new dataset requires only adding a new parser file. This facilitates the handling of multiple datasets and file formats within a single platform. OdoViz can also be equipped with extensions that allow inspecting and modifying loaded odometry data, reading external files, comparing images, etc.

- **Real-time:** Data is processed and visualised in real-time. Powered by JavaScript and React/Redux, OdoViz has asynchronous execution and immutability at its core, i.e., it runs tasks asynchronously without blocking the UI, updates the data in an immutable fashion by updating existing pointers to point to the new data, and displays the changes reactively. This pipeline allows the changes to variables to reflect on the visualisation near instantly.

Figure 3.13 shows the architecture of OdoViz with the *thin-server* and *rich-client* design. The front end has a Redux *store* to manage the app's *state* in a nested object literal, which acts as a single source of truth for all of the app's data and configuration. The *store* is read-only and can only be immutably modified as the user interacts with the app using *reducers*. *Reducers* are pure functions that take in the *previous state* and the current *action* to produce a *new state*. The reactive UI comprises a collection of components, each of which subscribes to required *store* data to populate its view template. When events occur, the UI dispatches *actions* that trigger the corresponding *reducer* functions. As a result, the *store* data changes, and all subscribed components

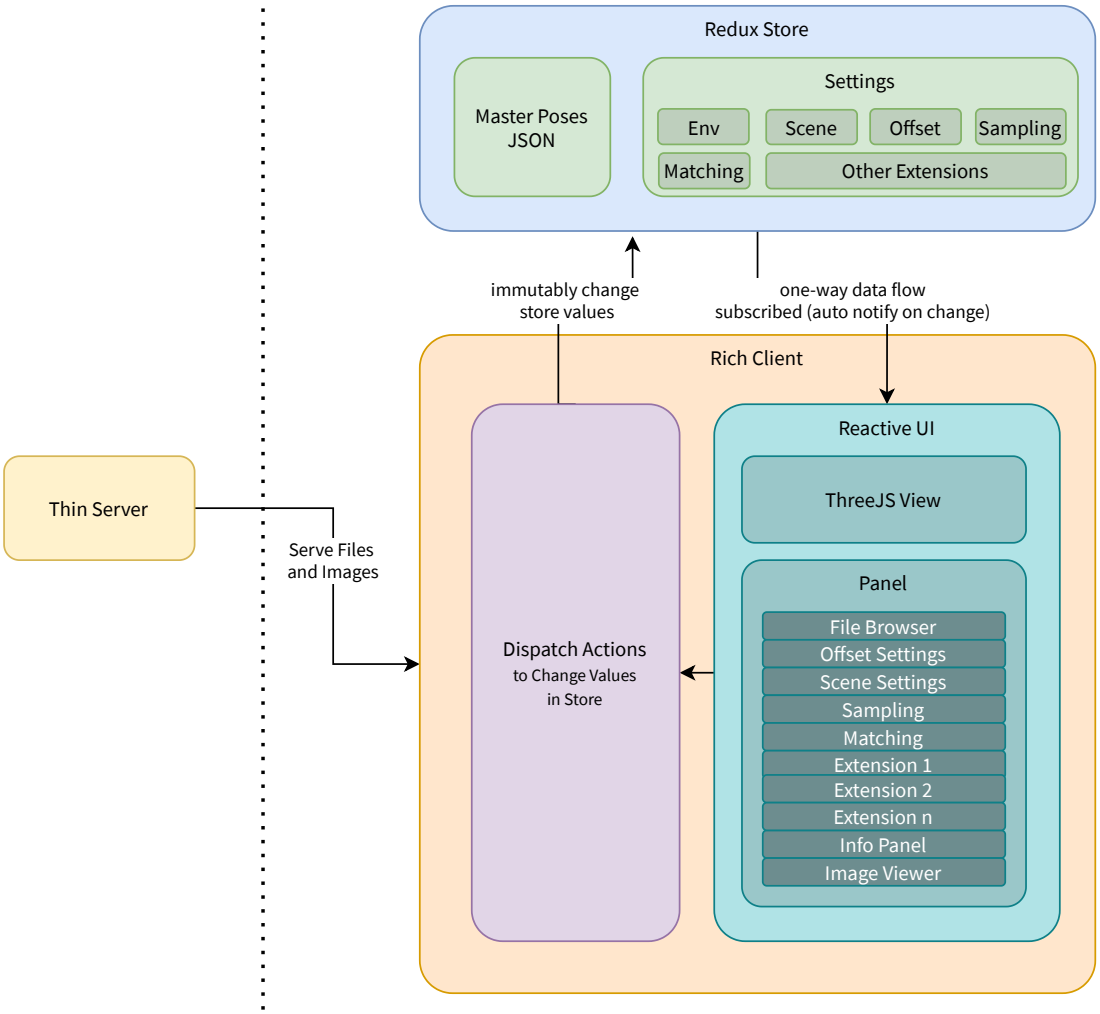


Figure 3.13: Overall architecture of OdoViz

re-render to reflect the updates. This unidirectional data flow ensures that the app is more predictable, traceable, and reproducible.

The OdoViz architecture allows the system to be deployed in a number of different ways, including:

- a browser-based frontend tool consuming data served on the same computer using a NodeJS server.
- a standalone single-user installation software packaged up individually for various operating systems using a packager system such as ElectronJS.
- an in-house app in an organisation or research group with a common server to serve files stored centrally, which can then be consumed by its members.
- an online interactive visualisation tool for publicly available datasets that will allow the users to visually explore and inspect the various trajectories before downloading them. With all the computation taking place in the client, this allows setting up a server with virtually no compute load, hosted only to serve files. For servers with limited network bandwidth, a smaller subset of the dataset can be made available to be interactively viewed. The software can additionally be customised to have only limited features and can also be set up to allow users to selectively download desired trajectories.

3.4.2 Core Modules

In this section, details are provided for each of the core modules that make up the overall system.

3.4.2.1 Data Parser

OdoViz supports loading Oxford RobotCar [7] GPS, INS, and Odometry *csv* files, BDD100K [134] *json* files, CMU Seasons [43] *nvm* files, KITTI [126] *txt* files, New College [110] *txt* files, and St Lucia [136] *log* files. OdoViz also supports loading other generic *.nvm* bundler files with point-cloud visualisation. Additional parser files can

be easily added to provide support for custom datasets. Data parsed from any source is consistently loaded into the browser's memory in the app's *state*, containing the following keys: *index*, *timestamp*, *position*, *orientation*, *gps*, *altitude*, *imageIndex*, and *image*. Visualisation, sampling, matching, and all other tasks can then use this data for processing. In cases where it may not be possible to load proprietary file formats, the files must first be converted to an *open format* and then parsed accordingly. For example, *.mat* files, which are binary MATLAB files that store workspace variables, should be loaded in MATLAB and then exported as one or more *.csv* files.

The fused GPS/INS will often have pose data with a frequency over 50 Hz, while the camera will typically operate at a lower frequency. Furthermore, in general, the GPS/INS timestamps will not be synchronised with the camera images, as the sensors record the information independently. It is therefore preferable to only compute information necessary for the poses from which the images are captured. This pose information is computed by the data parser by interpolating the neighbouring poses from the GPS/INS data. The included data parsers employ linear interpolation between the two nearest poses; however, this can be updated as necessary within any given parser. Poses are coloured using a gradient (red to orange by default), making it easy to distinguish poses that belong to overlapping traversals (see Figure 3.14).

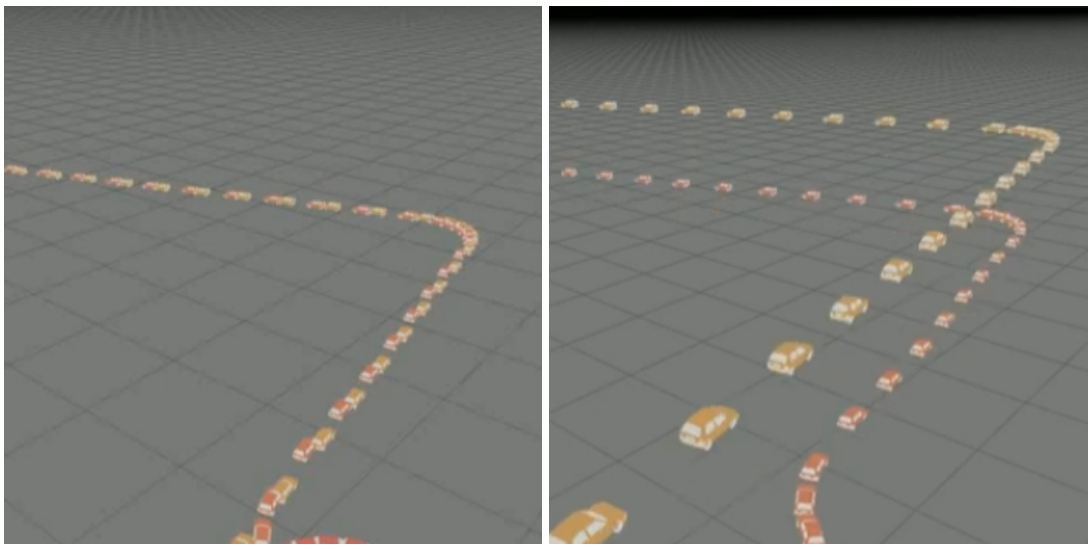
Performance Optimisations: Routine data manipulation operations, including map, reduce, and filter, can be efficiently implemented in JavaScript, often without requiring extensive language-specific knowledge. However, the custom parsing and matching scripts in OdoViz are executed in a web client, which imposes certain processing limitations. In particular, the JavaScript engine within the web browser cannot fully utilise all the cores of the CPU for compute-intensive tasks, and support for general-purpose GPU computation is severely restricted. To mitigate these limitations, asynchronous operations such as data fetching and parsing are delegated to separate worker threads using the Web Workers API. Additionally, libraries like GLMatrix [172] can be employed to accelerate matrix operations by leveraging WebGL, enabling partial use of the client-side GPU. Alternatively, compute-intensive operations and GPU-based processing can be delegated to the server. To do this, OdoViz's NodeJS backend can be extended to expose API endpoints that invoke native server-side code, enabling seamless integration with the frontend. It is worth noting that WebAssembly (WASM) can offer higher raw compute performance in the browser. However, compiling from languages such as Python to WASM can be non-trivial and may even result in reduced performance in

practice. This server-based approach instead facilitates the use of established scientific computing libraries such as NumPy [173], SciPy [174], OpenCV [175], and Eigen [176] while maintaining native performance and a modular, maintainable codebase.

3.4.2.2 Offset Settings

Many datasets and their associated file formats will have their own conventions for coordinate frames, e.g., swapped Y and Z axes for positioning and inverted Y rotations. OdoViz allows adjusting the offsets in each of the 3 dimensions for position P_x, P_y and P_z and rotation R_x, R_y and R_z. Functionality is provided to additively invert values assigned to each of the above six variables, swap any two R axes or P axes, and/or scale up or down all axes equally. With known translation and rotation of the camera, it is possible to precisely visualise the vehicle's pose using these offset settings. Furthermore, this conveniently allows loading unadjusted data captured from cameras or INS sensors mounted upside-down, rotated, and/or translated, and thus is also helpful in making new datasets from captured raw logs. The user interface is carefully designed to allow angle snapping to multiples of 45° and provide controls for fine-tuning to precisely adjust the scale.

Given the increased error in GPS altitude data [177, 178] when compared to latitudinal and longitudinal data, loading a journey with the same start and end points can result in a significant error in the z-coordinate (e.g., the Oxford RobotCar 2014-12-10-18-10-50 trajectory exhibits a significant altitude drift). To address this, OdoViz includes an option to ignore altitude during visualisation. Additionally, the system repurposes the unused z-axis to represent time differences. In addition to gradient colouring, the poses are elevated along the z-coordinate based on their timestamp, allowing a visual representation of temporal progression. With the rate of elevation being user-configurable, this feature is particularly useful when visualising multiple and overlapping journeys and in developing extensions and plugins to visualise loop closures, such as the HTMap [117] extension (explained later in Section 3.4.4.3).



(a) (b)

Figure 3.14: Screenshots showing the difference between flattened and time-encoded z-axis. (a) Flat z-axis, only coloured based on the index of the pose; (b) Time-encoded z-axis, z-offset added based on the index of the pose. As it is in 3D, the difference is understood more clearly in the video available at https://youtu.be/KsksVkYRmlg.

3.4.2.3 Scene Settings

By default, the 3D scene is set up with an oblique view, directional light, and an auto-expanding grid for reference, with each vehicle pose represented as a low-polygon 3D model of a car. Further settings are provided to switch between different preset viewpoints (e.g., top view), toggle the grid and lights, and adjust the scale of the elements of the 3D model to cater for different levels of zoom.

Similarly, for data with point clouds, the size of the points can be adjusted for either the selected pose or the entire dataset. Point-clouds are uniformly coloured with perceptually uniform[8] *viridis* colourmap [179] based on the depth using a GLSL shader. User-defined colourmaps can also be added.

Given that the noise associated with point clouds often increases significantly with depth, these areas consume a disproportionately large region of the colour spectrum, leaving a smaller bandwidth for the nearby points. To ameliorate this issue, this colour mapping can be adjusted by excluding farther points above a predefined percentile while

[8] if the data goes from 0.1 to 0.2, this should create about the same perceptual change in colour as if the data goes from 0.8 to 0.9

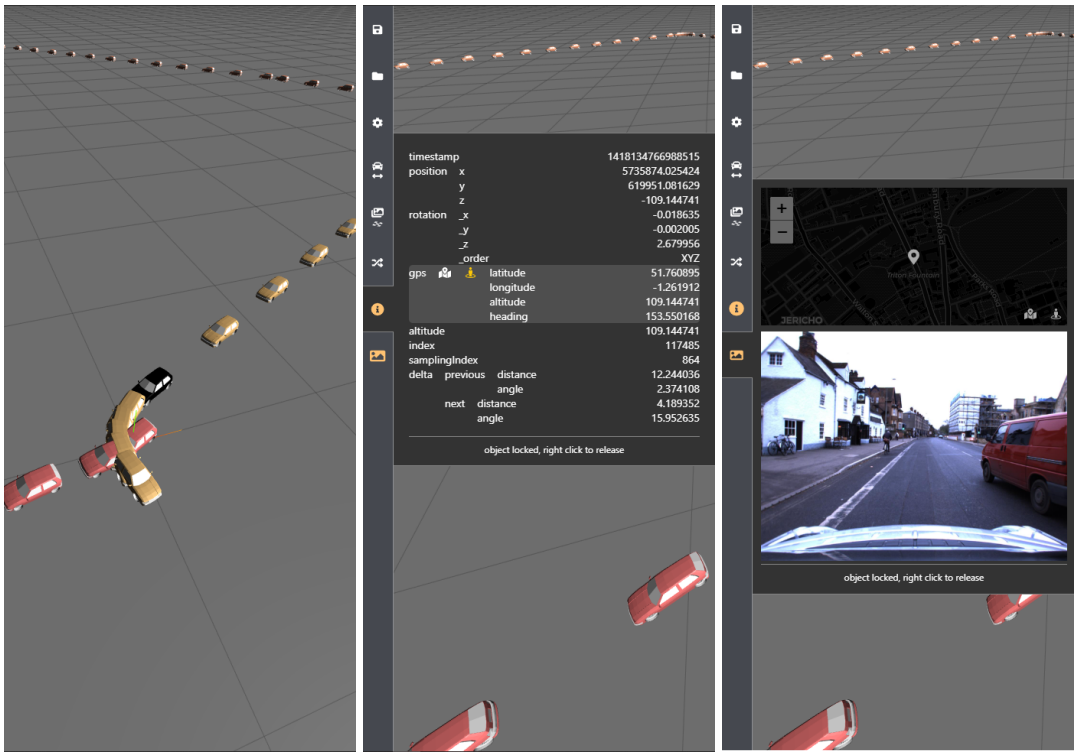
colouring. By default, we exclude points above the 90^{th} percentile. OdoViz further provides a setting to add a camera object to each pose, introducing new possibilities for extensions and plugins to add new tasks based on these cameras. For example, we have employed these camera objects to develop a plugin that finds intersecting camera frustums to filter images from overlapping viewpoints.

Mouse interaction to control the viewfinder defaults to *Orbit* controls (mouse-drag controls rotation and Alt + mouse-drag controls translation); however, this can be changed to *Map* controls (vice versa). Mouse-hover highlights and selects the pose, while right mouse-click pins/unpins the currently selected pose. Mouse-hover will not trigger selection when an object is pinned. This is useful when performing other operations on the scene while keeping the desired object selected.

Additionally, there is an animate feature that moves the view to the first pose and propagates to subsequent poses with a user-defined time delay to smoothly visualise a replay of the entire journey sequentially. As the animation proceeds, poses are selected one after the other, with the viewfinder's target set to the selected object and the selection being pinned to avoid other mouse interactions. This allows rotating and zooming in/out of the map whilst keeping the currently animated object in the centre of view.

3.4.2.4 Info and Image Viewer

On selection of a pose in the visualisation tool's main viewport (see Figure 3.15a) the associated information and the corresponding image can be viewed in the info panel and the image viewer panel, respectively (i.e., when the panels are activated). This information includes data such as latitude, longitude, altitude, and heading (see Figure 3.15b), and the image taken from the selected pose (see Figure 3.15c). We also developed a plugin to conveniently visualise in real-time the selected pose on satellite imagery using its GPS data, if available. To do this, we show the selected pose on a mini-map above the image using LeafletJS [180] and OpenStreetMap [181], which updates as the selection changes. Additionally, we integrated a feature that allows viewing the selected pose in a new browser tab on *Google Maps* and on *Google Street-View*. It should be noted that heading information of the current pose is used for comparison of the image against recent 360 captures from the same pose.



(a) Selected Pose (b) Info Viewer (c) Image Viewer

Figure 3.15: Screenshots showing (a) the viewport showing the selected pose highlighted in black, (b) information about the selected pose in the Info Panel, and (c) the image along with an embedded minimap in the Image Panel. Best viewed in colour on a computer screen.

3.4.3 Pose Processing

3.4.3.1 Adaptive Sampling

Many datasets offer dense data with poses sampled at a rate that can be as high as 200 Hz. For example, visualising a 10 km journey through Oxford[9] with GPS logged at 50 Hz yields 118,763 poses, whereas images logged at 16 Hz yield 35,514 poses. Rendering such a large number of data points would require excessive computational resources, resulting in very large loading times of up to a few minutes even on a modern high-end consumer-grade CPU. This data is often uniformly sampled using scripts to reduce the number of data points rendered in the 3D environment.

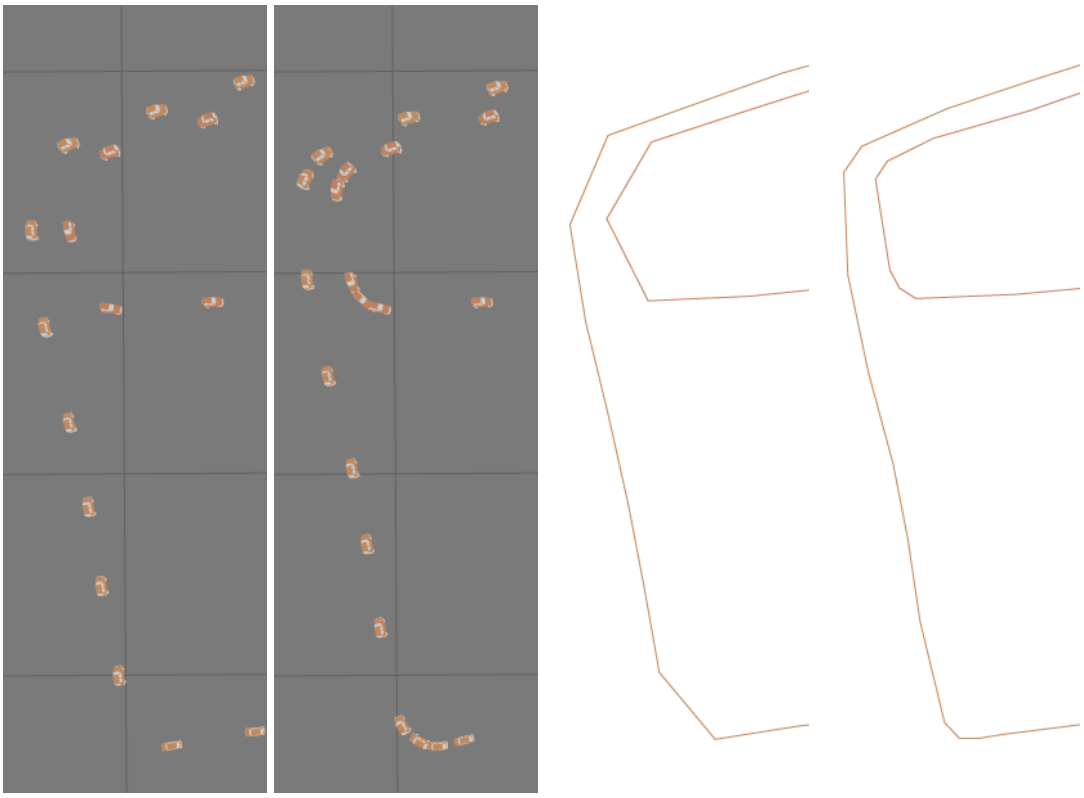
This uniform sampling is often also applied when training models that make use of keypoints in the images. In each of these cases, and particularly in the latter, such an approach to sampling may not be suitable. This is because many data points will be decimated around corners, despite exhibiting significant variations in visual content due to rapid changes in heading.

Usually a sparse set of data points are uniformly sampled from such dense data based on a fixed distance threshold. For example, one pose every 5 metres would reduce the total number of poses from 35,514 to less than 1000 in the above RobotCar trajectory. As such, this distance-based uniform sampling also removes redundant poses at the same GPS coordinates captured as the vehicle waits at a red traffic light at an intersection. However, we also lose many visually dissimilar and feature-rich images around the corners as the vehicle sweeps a larger heading angle in a very short distance. For example, in a tight turn with 10 or more visually distinct viewpoints, uniform sampling will reduce the resultant segment to one or two images if sampled only based on a distance threshold.

In order to overcome this issue, we present an adaptive sequence-based sampling technique that is dependent on the rate of change of angle with respect to distance. The algorithm for adaptive sampling is shown in Algorithm 1.

This technique traverses the poses using timestamps and decimates the poses only along the sequence, preserving the poses on overlapping routes within the journey. In

[9] Oxford RobotCar [7] trajectory 2014-11-18-13-20-12



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

Figure 3.16: Screenshots (a) and (b) show poses present in a portion of the Oxford RobotCar trajectory after uniform sampling and adaptive sampling, respectively. Contours connecting the sampled poses (c) and (d) highlight the difference in results between the two methods and show how the adaptive sampling preserves the corners of the trajectory. A screen-cast showing how changing the constraints $\tau_{d_{\text{acc}}}$ and $\tau_{\theta_{\text{acc}}}$ affects sampling in real-time is available at https://www.youtube.com/watch?v=9Vf26sgRqSc

Algorithm 1: Adaptive Sampling

Result: List of sampled_poses
poses = list of poses; // population, input
sampled_poses = []; // result placeholder
$d_{\text{acc}} = 0$; // accumulated distance
$\theta_{\text{acc}} = 0$; // accumulated angle
$\tau_{d_{\text{acc}}} = 12$; // static distance threshold (m)
$\tau_{\theta_{\text{acc}}} = 15$; // adaptive distance threshold (**deg**)
foreach *pose in poses* **do**
    $d_{\text{acc}} = d_{\text{acc}} + \Delta d$;
    $\theta_{\text{acc}} = \theta_{\text{acc}} + \Delta\theta$;
    if $\theta_{acc} > \tau_{\theta_{acc}}$ *or* $\theta_{acc} > \tau_{\theta_{acc}}$ **then**
        Add pose to sampled_poses; // choose sample
        $d_{\text{acc}} = 0$; // reset accumulated distance
        $\theta_{\text{acc}} = 0$; // reset accumulated angle
    end
end

contrast, binning the poses based on the GPS locations using a KD-tree and selecting one pose per bin would yield fewer samples, as all poses from a certain GPS location would fall in the same bin regardless of whether it belongs to an overlapping route or not. Hence, the sequence-based sampling is preferred, as the resultant set of poses can be used in many place recognition-related tasks, such as computing loop closure detection on fewer samples from the same trajectory.

A plugin for sampling added to the visualisation tool allows us to perform both uniform sampling using a static distance threshold $\tau_{d_{\text{acc}}}$, and adaptive sampling using an adaptive distance threshold $\tau_{\theta_{\text{acc}}}$ that is based on the angle accumulated θ_{acc} over a distance d_{acc} . We avoid computing interpolation of data as $\tau_{d_{\text{acc}}}$ or $\tau_{\theta_{\text{acc}}}$ changes by precomputing the interpolated data for all images. This makes it possible to interactively vary both $\tau_{d_{\text{acc}}}$ and $\tau_{\theta_{\text{acc}}}$ for adjusting sampling with real-time visual feedback.

Figure 3.16a and Figure 3.16b show the different poses sampled based on the uniform sampling and adaptive sampling, respectively. The plugin also shows total poses that will be sampled based on the criteria chosen and features a facility to export sampled poses to a *JSON* file that can be processed using a dataloader for training deep learning models.

3.4.3.2 Finding Pose Correspondences

Training computer vision models to extract features from images robust to different viewpoints, seasons, and different times of the day requires thousands of image pairs from corresponding locations. This data can be curated based on the GPS data. In curating such datasets, it is important to compare images and information regarding the poses of the same or another journey traversed on the same route, either partially or completely. We compute a matching pose with the least loss for each of the poses in the journey selected for finding correspondences, where the matching loss between a query pose p_x and a match candidate pose p_y is defined as follows:

$$\begin{aligned} \text{Loss} &= \alpha\, \Delta d + \beta^*\, \Delta\theta \\ \beta^* &= \begin{cases} \beta\, \dfrac{\theta_{\text{acc}}}{\tau_{\beta_\theta}}, & \text{if } \theta_{\text{acc}} > \tau_{\beta_\theta} \\ \beta, & \text{otherwise} \end{cases} \end{aligned} \tag{3.1}$$

where,

α = distance importance factor
β = angle importance factor
Δd = absolute distance[10] between p_x and p_y
$\Delta\theta$ = absolute heading difference between p_x and p_y
β^* = adaptive angle importance factor
θ_{acc} = angle accumulated up to a distance of τ_{β_d} from p_x
τ_{β_θ} = beta limiter threshold

β^* increases β by a factor proportional to θ_{acc}. As θ_{acc} increases when p_x is closer to corners, β^* dynamically raises the weight assigned to the angle loss when computing for query poses near corners, increasing the importance of angle when finding matches.

In order to speed up the matching process, we discard poses that have a distance loss greater than 30 m and any loss greater than a defined τ_{loss}. To avoid having multiple poses matching to more than one query pose, an additional step is performed to check if the match has been paired with any other pose. We update the match only if it has a lower loss than other matches. All the above operations are fully customisable to suit individual needs in a separate *matcher* file.

When a journey is loaded and sampled, OdoViz can load another overlapping journey and find matching poses for each of the poses in the current journey, as described above. The matched poses from the new journey are grouped together and added to the same 3D scene with a different colour. Figure 3.17 shows the colour-coded matching results of loading two traversals against the loaded traversal. As with the pose sampling plugin, this plugin features an export as *JSON* option where the resulting file can be used directly to train deep learning models, e.g., CNN-based metric learners [62, 63, 64].

3.4.4 Extensions and Plugins

Further to the core modules described in the previous sections, OdoViz's functionality can be extended through the addition of visualisation and compute extensions and plug-ins for importing and exporting data. Compute-intensive extensions are recommended

[10] computed using the Haversine Formula [182], i.e., the great-circle distance between two points on a sphere given their longitudes and latitudes

Figure 3.17: Left: the main viewport loaded with three Oxford RobotCar trajectories obtained by matching Winter Night 2014-12-10-18-10-50 (green) and Summer Day 2015-05-19-14-06-38 (indigo) against an adaptively sampled Winter Day 2014-12-09-13-21-02 (orange). Right: Images corresponding to the three matched poses marked in red in the left image: top – Winter Day, middle – Winter Night, and bottom – Summer Day.

to use Web Workers for compute operations that asynchronously load the compute data. Here we highlight three such extensions that are included with the software in order to provide examples of the versatility of the system.

3.4.4.1 Save and Restore Plugin

The Save plugin is a simple import/export plugin that saves *store* data to the browser's storage. The current view, offset settings, scene settings, sampling settings, and the loaded file can be saved and restored using this plugin. Additionally, the plugin indicates if there have been any changes made since the last load, for example, to check if the sampling was performed using previously saved settings before exporting sampled poses.

3.4.4.2 Top-k Image Retrieval Analysis Extension

A common approach for VPR using deep neural networks is to compute a compact embedding that provides a compressed representation of an image's visual appearance suitable for matching and retrieval. Often a top-k precision and/or top-k recall metric is used to evaluate the quality of the embeddings using retrieval performance. The *top-k Image Retrieval Analysis* extension accepts one or more (i) *.npz* files containing pairwise distances, labels, and top-k distances output during training for different epochs of a deep learning algorithm, and (ii) the corresponding *data.json* containing the mapping from label and index (or path) to the original image file location on disc. This data is then presented in an intuitive tabulated format showing top-k matches $k = 5$ by default. The interface allows the user to interactively explore the results, selecting different query or anchor images, visualising the top-k matches, varying k using a slider, etc. (see Figure 3.18). In particular, when a row is selected in the table, an image comparison interface on the right shows images of the ground truth and the top match, one below the other, while showing smaller thumbnails of the top-5 matches below. Ground Truth is shown in yellow, while the match is shown in blue — represented as a font colour in the table and as a border in the image comparison interface. Results for a given epoch can be compared with other epochs using the slider provided. We extensively use this extension to compare and qualitatively evaluate top-k retrievals of various image retrieval models later in Chapter 4, Chapter 5, and Chapter 6.

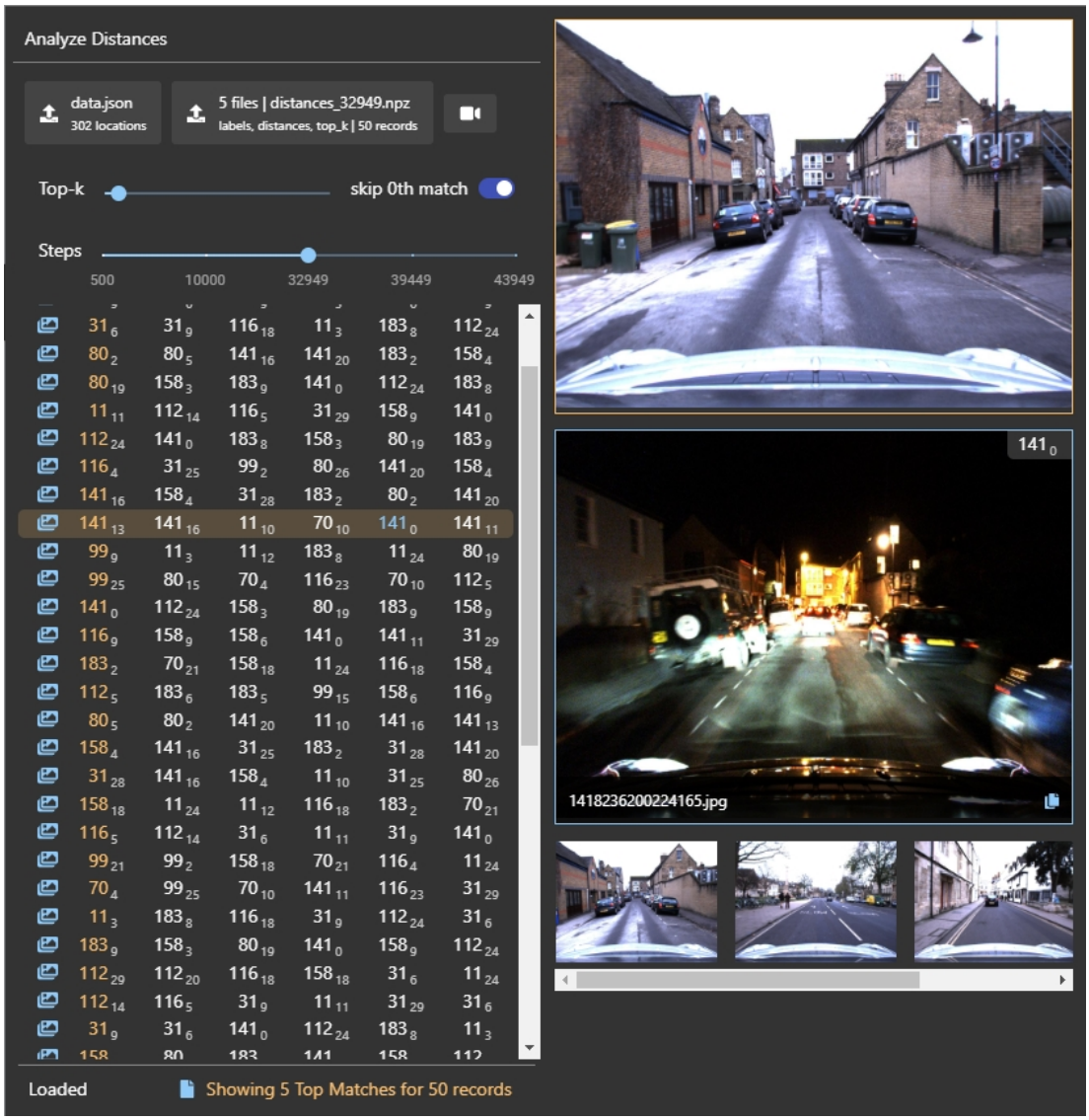


Figure 3.18: Screenshot of the Top-k Image Retrieval Analysis extension. On the left, the interface displays a table corresponding to the $32,949^{\text{th}}$ batch of training data, with $k = 5$ selected via the *Top-k* slider. The highlighted row indicates the selected anchor image and its top-k matches. On the right, the anchor image is shown at the top, and the top-k retrieved matches are presented as thumbnails in a horizontally scrollable view at the bottom. Selecting a match from the thumbnail view or the table displays it in a larger format below the anchor image for direct visual comparison.

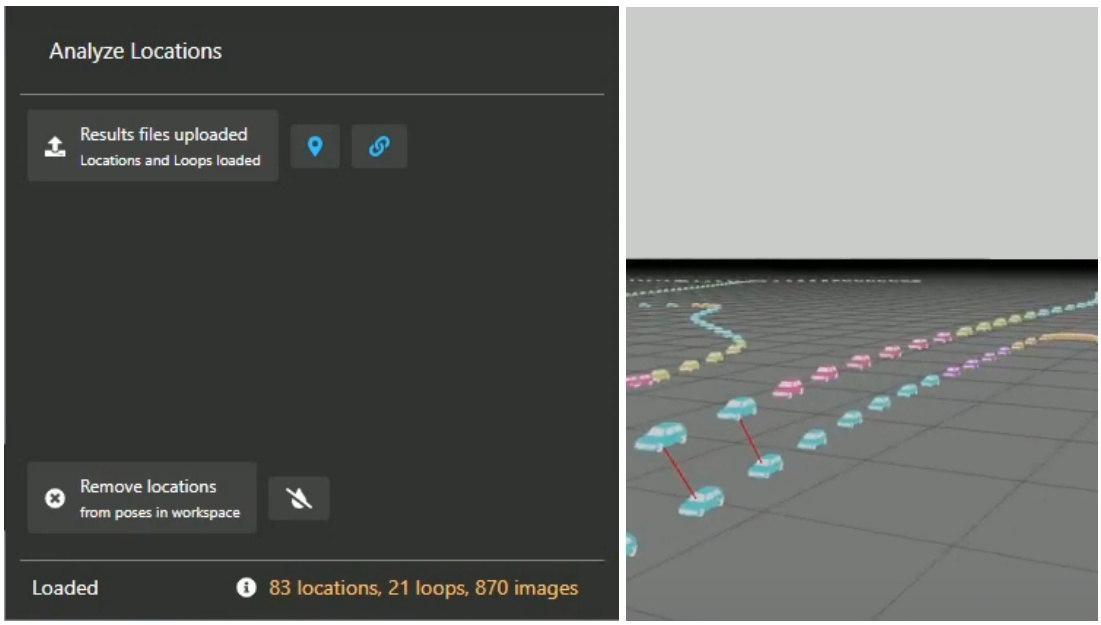


Figure 3.19: Screenshot of HTMap plugin showing loop closure connections in an overlapped route within the same trajectory loaded with time in Z-axis

3.4.4.3 HTMap Extension

As a final example, we present a visualisation plugin to render the output of the Hierarchical Topological Maps (HTMap) technique of Garcia et al. [117]. This approach divides trajectories by hierarchically grouping images into a set of topologically connected nodes. The HTMap extension allows loading results of HTMap to provide a 3D view of how the trajectory is divided into multiple parts with a different colour for each location and where the image loops have been found. Figure 3.19 shows the HTMap result for one of the Oxford RobotCar trajectories. Notice the different colour for sets of poses belonging to the same topological node and the loop closure connections in red that connect the poses of the images matched. We make heavy use of this feature in visualising and interpreting the results presented in Chapter 6.

3.5 Metrics

In the field of VPR, the effectiveness and reliability of different approaches are measured using a variety of metrics. To effectively evaluate VPR systems that use feature representations for image processing and operate on sequential image sets for mapping,

it is essential to adopt a comprehensive suite of metrics tailored to meet these specific prerequisites. This section delineates various metrics, encompassing both quantitative and qualitative aspects, vital for benchmarking VPR systems.

To understand the metrics better, it is essential to consider the elements of a confusion matrix:

- **True Positives (TP)**: Correctly identified matches or loops.
- **True Negatives (TN)**: Correctly identified non-matches.
- **False Positives (FP)**: Non-matches incorrectly identified as matches.
- **False Negatives (FN)**: Actual matches missed by the model.

Additionally, the following terms are crucial for a comprehensive understanding:

- **Predicted Positives (PP)**: Matches predicted as positive, $PP = TP + FP$.
- **Predicted Negatives (PN)**: Matches predicted as negative, $PN = TN + FN$.
- **Positives (P)**: Total positive matches, $P = TP + FN$.
- **Negatives (N)**: Total negative matches, $N = TN + FP$.
- **Total Population**: The sum of all positives and negatives $= P + N = PP + PN$.

3.5.1 Accuracy, Precision, Recall, and F1 Score

The most widely used performance metrics in classification and retrieval tasks — accuracy, precision, recall, and F1 score — are also directly applicable to evaluating VPR systems.

Accuracy measures the proportion of correct predictions among all predictions:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} = \frac{TP + TN}{P + N} \tag{3.2}$$

While intuitive, accuracy alone may not reflect model performance in the presence of class imbalance, which is common in VPR tasks.

Precision (Positive Predictive Value) quantifies the proportion of true positive matches among all predicted positives:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{TP}}{\text{PP}} \tag{3.3}$$

Recall (Sensitivity or True Positive Rate) measures the proportion of actual positives that are correctly identified:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\text{P}} \tag{3.4}$$

The **F1 score** is the harmonic mean of precision and recall, providing a balanced single metric that considers both:

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.5}$$

These metrics are well established; see [183] for a comprehensive discussion.

3.5.2 Recall at 100% precision

In SLAM systems, VPR modules must achieve extremely high precision, as any incorrect loop closure (FP) can corrupt the map, causing substantial errors or complete mapping failure. By contrast, missed loop closures (FN) are less critical, though they still impact performance.

Therefore, **recall at 100% precision**, which measures the proportion of true loop closures detected when no false positives are allowed, is used as a key metric. This metric directly reflects a VPR system's suitability for integration into SLAM, where maintaining map integrity is paramount. High recall at 100% precision indicates that the system can reliably detect all relevant loop closures without risking erroneous associations.

3.5.3 ROC Curve, PR Curve, ROC AUC and AP

Precision, recall, and the F1 score all depend on the choice of a decision threshold, which can significantly affect reported performance. To address this, threshold-independent metrics are widely used in VPR evaluation.

The **Receiver Operating Characteristic** (ROC) curve illustrates the trade-off between the True Positive Rate (TPR, or Recall) and the False Positive Rate (FPR) across all possible thresholds, with the area under the ROC curve (ROC AUC) providing an overall summary of discrimination ability. The FPR quantifies the proportion of negative cases incorrectly classified as positive.

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} = \frac{\text{FP}}{\text{N}} \tag{3.6}$$

Similarly, the **Precision-Recall** (PR) curve plots precision against recall at different thresholds, and its summary metric, average precision (AP), is particularly informative in the presence of class imbalance. Together, these metrics offer a more comprehensive and robust assessment of VPR systems, independent of any specific threshold setting.

3.5.4 Top-k Metrics

Top-k metrics are designed to evaluate the model's performance in retrieving the correct matches within its top-k predictions. In the evaluation of VPR systems, especially those designed for navigating complex environments or extensive databases, the top-k metrics play a pivotal role in assessing the system's ability to accurately identify and rank the most relevant matches within the top-k results of a query. Given their significance in applications where immediate and precise location identification is crucial, top-k metrics provide valuable insights on a VPR system's practical effectiveness. They specifically evaluate how well the system prioritises potential matches, which is essential for real-time navigation and associated image retrieval and matching tasks.

Recall@k, or top-k recall, evaluates the system's ability to include at least one match within its top-k predictions. Thus, it reflects the coverage of the system in ensuring that the correct match is not overlooked in the initial ranked set.

Precision@k, or top-k precision, conversely, measures the proportion of relevant images among the top-k predictions. It gauges the system's effectiveness in ranking the most relevant locations higher.

Formulas for recall@k and precision@k are given by:

$$\text{recall@k} = \frac{1}{N}\sum_{i=1}^{N} \min(\left|\text{top}_k(\text{PP}_i) \cap \text{TP}_i\right|, 1) \tag{3.7}$$

$$\text{precision@k} = \frac{1}{N}\sum_{i=1}^{N} \frac{\left|\text{top}_k(\text{PP}_i) \cap \text{TP}_i\right|}{k} \tag{3.8}$$

where,

N = number of query images
TP_i = set of True Positives for i^{th} query image
PP_i = set of Predicted Positives for i^{th} query image
$\text{top}_k(\text{PP}_i)$ = top k matches based on similarity score from PP_i

A predicted match for a query image is classified as a True Positive (TP) if it falls within a predefined distance threshold from the query image's ground truth GPS coordinates. In the context of VPR systems, the recall@k metric is often employed, where a query image is considered correctly localised if at least one of the top k retrieved database images is within a predefined distance (e.g., 25 m) from the ground truth position of the query. The proportion of correctly recognised queries is then calculated for various values of k, making recall@1, the most stringent criterion, a rigorous measure of the model's performance.

In retrieval systems, these metrics are essential in contexts where the order of predictions is important. The recall@k metric is particularly useful, as missing relevant images within the top-k predictions leads to missing potential loop closures. Thus, recall@k and precision@k metrics provide insights into system efficacy by balancing between relevance and comprehensiveness within a limited result set.

3.5.5 Cluster-based Metrics

In VPR systems, evaluating the quality of feature embeddings is a critical step for ensuring the effectiveness of the system. Cluster-based metrics evaluate the quality of feature embeddings using clustering techniques. Cluster-based metrics allow for an indirect assessment of the inherent structure of the data captured by the embeddings. By evaluating how well the embeddings can be clustered, we gain insights into whether the embeddings have captured meaningful patterns and distinctions present in the

visual data. Hence, this assessment is crucial even in scenarios where clustering is not explicitly employed as an objective during training.

Normalised Mutual Information (NMI) is a measure used to assess the quality of clustering. It quantifies the mutual dependence between the clustering results and the true labels, normalised to account for the size of the clusters. Each image descriptor is assigned to one of the K clusters, where K is the number of locations (or regions) from which the images are sampled. The assigned cluster indices are then compared against the ground truth location indices to obtain an NMI score. A higher NMI indicates better clustering quality. NMI is given by the following expression:

$$\text{NMI} = \frac{2 \times \text{I}(Y; \hat{Y})}{\text{H}(Y) + \text{H}(\hat{Y})} \tag{3.9}$$

where,

$\text{I}(Y; \hat{Y})$ = mutual information between assigned cluster labels Y and true labels $\hat{Y}$
$\text{H}(Y)$ = entropy of assigned cluster labels Y
$\text{H}(\hat{Y})$ = entropy of true labels $\hat{Y}$ and

$$I(U, V) = \sum_{u \in U} \sum_{v \in V} P(u, v) \log\left(\frac{P(u, v)}{P(u)P(v)}\right) \tag{3.10}$$

$$H(X) = -\sum_{x \in X} P(x) \log P(x) \tag{3.11}$$

NMI can further provide an indication of how well the system might generalise to unseen data. Good clustering, reflected in high NMI scores, implies that the model has learned robust features that can potentially categorise new, unseen locations accurately.

Furthermore, **Cluster Cohesion** and **Cluster Separation** provide qualitative measures of intra-cluster and inter-cluster distances, respectively. Cohesion evaluates how close the elements of a cluster are to each other, ideally indicating tight, well-defined clusters. Separation assesses how distinct or separate different clusters are from one another, which is crucial in ensuring that different locations or images are not erroneously grouped together. These metrics are vital where spatial relationships and scene similarities play a significant role in the recognition process. Hence, these metrics

are useful in determining how well the feature embeddings can discriminate between different locations or scenes.

Assessing the embeddings with these metrics can further guide the tuning of the feature extraction process. A lack of cluster cohesion in the embeddings, for instance, could suggest that the model's architecture or training procedure requires adjustment such that the features of similar images are more closely aligned in the embedding space.

In a comparable fashion, significant effort ought to be given to analysing the balance between capturing the general characteristics of a place (for instance, all images of a park) and recognising specific features (like a particular statue in the park). This balance is critical for ensuring that the VPR system is versatile enough to identify a location broadly while also being sensitive to specific elements that differentiate one place from another. Cluster-based metrics help in understanding this diversity-specificity balance, ensuring that the embeddings are neither too generic nor overly specific.

3.5.6 Runtime, Compute, Efficiency, and Scalability

While the previously discussed metrics address key aspects of robustness and invariance in VPR systems, additional considerations such as runtime, computational resources, search efficiency, and scalability are also critical for a comprehensive evaluation.

Runtime: A crucial factor for VPR systems, particularly in applications requiring real-time processing, is the total time taken for the entire mapping process. The Bag-of-Words (BoW) techniques and other approaches that incorporate keypoint-based localised descriptors often entail longer matching times due to the complexity of feature matching. In contrast, systems employing holistic representations can compare images more rapidly using faster distance functions such as Euclidean distance, cosine similarity, or Chi-squared distance for histogram-based descriptions. This approach significantly reduces the time required for image comparison, enhancing the system's overall runtime.

Efficiency: This metric encompasses various factors such as descriptor length and storage requirements, descriptor computational complexity, the mean number of images searched per query, and the system's ability to parallelise operations. For

example, a larger descriptor size increases storage demands and slows down comparison operations, while high computational complexity in descriptor generation and similarity measurements results in longer processing times. Efficiency can be improved by implementing more streamlined computational operations during descriptor generation, designing compact descriptors to minimise storage and processing requirements, and optimising descriptor compute time through efficient algorithms leveraging hardware acceleration.

Scalability: The scalability of a VPR system is measured by its ability to maintain performance with the increasing length of trajectories, diversity of scenes, and a growing database of previously seen images. A scalable system should perform comparably on extensive, diverse trajectories without fully expending the time for searching for matches. Indexing and hierarchical techniques also play a pivotal role in improving scalability. By structuring the database efficiently, these techniques facilitate quicker access to relevant data, even as the database expands, ensuring the system's capability to handle larger datasets without a proportional increase in computational demand.

The choice of image representation, search algorithms, and database management strategies plays a critical role in optimising these aspects, ultimately determining the system's suitability for real-world deployment.

3.6 Conclusion

In this chapter, we presented various VPR datasets, advanced and bespoke tools to utilise them, and a comprehensive set of metrics as foundational elements in developing and assessing VPR systems. We explored various metrics necessary to do an in-depth analysis of the strengths and weaknesses of a VPR system and for a rigorous evaluation of the same.

We discussed and reviewed various public VPR-specific datasets, including those that have trajectories traversed multiple times, wholly or in part, providing multiple sequences of images traversed at different times of day, weather, and/or seasons. We revealed various useful visualisation and odometry processing tools to consume and operate on the datasets discussed. We then walked through the steps and challenges involved in utilising these public datasets using various tools for VPR research. To

assist the research community in addressing these challenges and to support the research carried out within this thesis, we presented the OdoViz odometry processing framework. OdoViz provides a unified approach to visualise, analyse, interactively inspect, and curate data necessary for VPR research across a wide variety of heterogeneous benchmark datasets.

We discussed the popular metrics such as precision and recall while also discussing the significance of ROC and PR curves in assessing the performance of the VPR system under different decision threshold settings. We further discussed the threshold-independent VPR metrics such as ROC-AUC and AP. We then reviewed top-k metrics and their importance in retrieval tasks. Furthermore, we elaborated on the role of cluster-based metrics in assessing the quality of feature representations and their uses in training for better feature extraction. Finally, we highlighted the importance of runtime, efficiency, and scalability metrics that are useful in determining the system's suitability for deployment. The latter point is of added importance in this thesis, where in the following chapters we investigate a number of approaches to directly optimise for these metrics.

Chapter 4

Robust Learned Descriptors

4.1 Introduction

Although the traditional approaches [13, 39, 40] using the Bag of Visual Words (BoVW) approach permitted a reliance on visual place recognition within SLAM systems, they lacked the repeatability and robustness required to deal with the challenging variability in appearance that occurs in natural scenes caused by different times of the day, weather, lighting and seasons; see Figure 4.1.

With the advent of CNNs and their compelling results over traditional methods in tasks such as semantic segmentation and feature learning, researchers have sought to improve the robustness of VPR systems by using CNNs to incorporate semantic, geometric, and topological information from the scene. Within the domain of VPR, early learned approaches utilised ImageNet [96] pretrained AlexNet [54], VGGNet [57], and ResNet [58], without the last classificiation layer and obtained image embeddings by flattening feature maps from the last layer. Researchers also used CNNs pretrained for dense semantic segmentation for VPR. Segmap [185] uses CNNs to eliminate moving objects using semantics and to generate compact embeddings for 3D objects. LoST [85] uses *conv5* layer of modified dense semantic segmentation neural network RefineNet [86] to generate embeddings.

The validity of embeddings generated using pretrained CNNs has a significant dependency on the training data. For example, RefineNet was pretrained on the Cityscapes

Figure 4.1: Images of the same place taken at different times demonstrating the challenge of place recognition in the context of extreme changes in visual appearance. Image Credits: [184]

dataset [132] containing images from different cities; however, it does not contain sequences that were captured at different times of the day under varying weather conditions or scene types (e.g., rural). Hence, the network exhibits limited performance on scene conditions it has not previously seen. Retraining these networks with such images would be challenging, as it is a tedious and expensive task to obtain ground truth dense semantic segmentation labels.

To overcome such label constraints, researchers have sought to use metric learning with weak supervision where deep learning models were trained using contrastive losses[1] to implicitly encode semantic, photometric, and visual information from images without the need for ground truth labels [63, 67, 62, 66, 64]. In these approaches, the ML model maps input images to an embedding space that is optimised to minimise the distance between embeddings of the same place whilst maximising the distances between embeddings of different places. However, these techniques are not directly applicable to large sequential datasets due to data matching, mining, and pairing limitations. Furthermore, such techniques are prone to failure due to training instability caused by embedding collapse, embedding explosion, or stalled learning.

In this chapter, we address these issues and propose a set of techniques to train neural networks on large sequential datasets in a reliable and repeatable manner to output learned representations for VPR. We employ specific data sampling, data augmentation, and training techniques along with architectural and loss function adaptations to ensure stable and efficient training of ML models without compromising retrieval performance. In particular, we:

- propose a novel approach of discretising trajectories into regions called locations containing similar images from the same place to efficiently obtain unique triplets during training,
- employ adaptations to the loss function, architecture, and learning rate to mitigate training failures,
- propose to aggregate discretised locations combined with data augmentation techniques to add viewpoint variance without the use of additional images,
- build batches of training data in an online fashion and train the network progres-

[1] losses computed contrasting two or more data point representations

sively, by gradually increasing difficulty to avoid network collapse due to being unable to find correlations between the positive image pairs, and,

- employ computational and memory optimisations in our implementation for faster and more efficient training.

We present the results of training with weakly supervised data curated from large sequential public datasets utilising the proposed set of techniques and adaptations. As part of this, we also present the results of an ablation study to evaluate the impact of individual contributions on the overall performance.

Elements of this chapter have been published as *Place Recognition in Challenging Conditions* [186] at the 2019 Irish Machine Vision and Image Processing (IMVIP) conference.

4.2 Background

With the rapid development of deep learning, learned feature descriptors outperformed handcrafted descriptors, overcoming their limitations [188, 189]. Chen et al. [190] proposed a CNN-based place recognition method for the first time in 2014, demonstrating a significant increase in recall at 100% precision. Many early learned VPR approaches utilised ImageNet [96] pretrained AlexNet [54], VGGNet [57], and ResNet [58], without the last classification layer, and obtained image embeddings from the feature maps of the CNN's last layer [191, 192, 193].

Following this, researchers also employed CNNs pretrained on other tasks such as dense semantic segmentation and depth prediction that are better suited for VPR. In SegMatch [84], objects are segmented from LiDAR point cloud data accumulated for approximately one second, with the 3D object's description stored along with its semantic label. When a new frame arrives, moving objects are eliminated using their semantic labels, where the remaining objects are then compared to those from earlier segments using their 3D descriptors. In LoST [85], output tensors from one of the convolutional layers, *conv5*, of a modified version of the dense semantic segmentation neural network RefineNet [86], were used to generate a deviation from the mean tensor for three semantic labels — road, building, and vegetation — which were then flattened

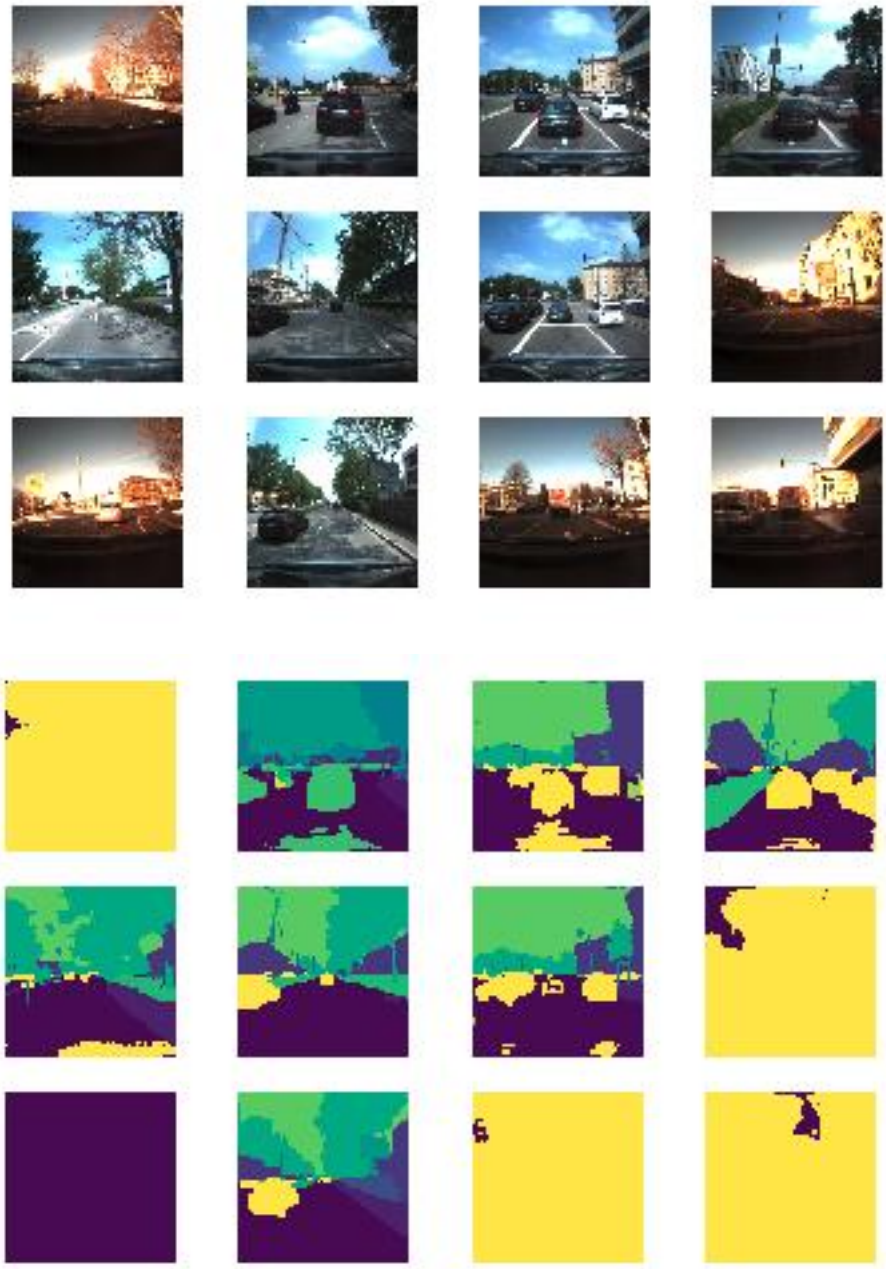


Figure 4.2: (Top) Input Images. (Bottom) Dense semantic segmentation results from PSPNet [187] pretrained on Cityscapes [132]. The network does not work as expected for images taken at different times of the day or during different seasons.

to an embedding. Embeddings were generated for each image during the traversal. Each time an embedding was stored, it was compared to existing embeddings, and the embedding distances were computed. Among the images with embedding distances below a defined threshold, the authors further refined the selection by matching the maximally activated regions.

We first tested several images taken at different time periods on state-of-the-art dense semantic segmentation networks. This was done to determine how reliably we can use the predicted semantic labels for place recognition. Pyramid Scene Parsing Network (PSPNet) [187], pretrained on Cityscapes [132], one of the top 5 dense semantic segmentation networks on the PASCAL VOC 2012 challenge [194], showed impaired semantic output when applied to challenging images (sunset/winter) compared to its output on sunny or overcast daytime images; see Figure 4.2. Similarly, RefineNet, the backbone network used to build embeddings in LoST, which was also pretrained on Cityscapes, was not able to produce reliable semantic labels for such challenging images. This can be directly attributed to the network not having seen such images during the training phase.

Retraining these networks to incorporate these types of images would be both tedious and expensive given the effort required to obtain ground truth dense semantic segmentation labels over large-scale datasets. To remove the direct dependency on manually annotated dense semantic labels, researchers have instead employed weakly supervised metric learning techniques [63, 67, 62, 66, 64] to train over image pairs (or triplets) of the same place taken at different times. FaceNet [63] demonstrated face recognition and verification through clustering employing a triplet loss. NetVLAD [67] builds upon VLAD, adapting it for deep learning-based VPR using weakly supervised learning without requiring explicit labels. Kim and Walter [66] used a neural network that matches ground-level images to satellite imagery trained with a pairwise contrastive loss using a Siamese network.

Although these approaches mark a significant shift from depending on fully human-annotated data labels, we find that these techniques are not directly applicable to training ML models on large sequential datasets. Many large sequential datasets, such as Oxford RobotCar [7], offer data with a higher sensor capture frequency, resulting in millions of individual image frames extracted from continuous driving videos. The sheer size of the dataset makes it impractical to search for and generate pairings (positive and negative)

for each image present on the fly during training. Furthermore, a significant portion of these frames contain redundant visual information, particularly in scenarios where the vehicle is stationary at traffic lights, intersections, or similar contexts. Utilising all of these frames would considerably extend training times without contributing substantial new visual information. In our work, we address these problems and propose a novel approach of discretising the trajectory into locations, from which images can be sampled during training time.

Datasets such as Freiburg Across Seasons [128] and Oxford RobotCar [7] contain many traversals of the same trajectory exhibiting long-term seasonal changes. However, the viewpoint variance exhibited across different traversals within these datasets is confined due to the limited number of lanes present in the chosen routes. To overcome this, we propose aggregation of locations to allow for positive samples from nearby locations and combining it with aspect ratio-preserving data augmentation techniques to train for greater viewpoint variances. More specifically, we reformulate commonly used translation, rotation, and crop augmentation techniques as custom image operations that do not distort, zero out, or pad parts of the training images.

Training deep learning models on large sequential datasets directly using contrastive losses, such as ranked pairwise loss and triplet loss, often proved intractable. This was primarily due to stalled learning or training failures, which hindered the model's ability to form meaningful representations. To address these challenges, we introduce adaptations to the loss functions, network architecture, and learning rate schedules, coupled with a progressive training strategy aimed at faster loss convergence and stable training. Additionally, compute and memory optimisations are incorporated into our implementations for greater efficiency. The underlying causes of these issues, along with a detailed explanation of the proposed solutions, are presented in the next section.

4.3 Methodology

In this section, we first explain the VPR pipeline that describes how a learned VPR model using existing approaches can be used in an SLAM system. In particular, we utilise the embedding approach, wherein we infer an embedding vector for each image using the CNN that implicitly captures shapes, edges, associations, gradients, etc.,

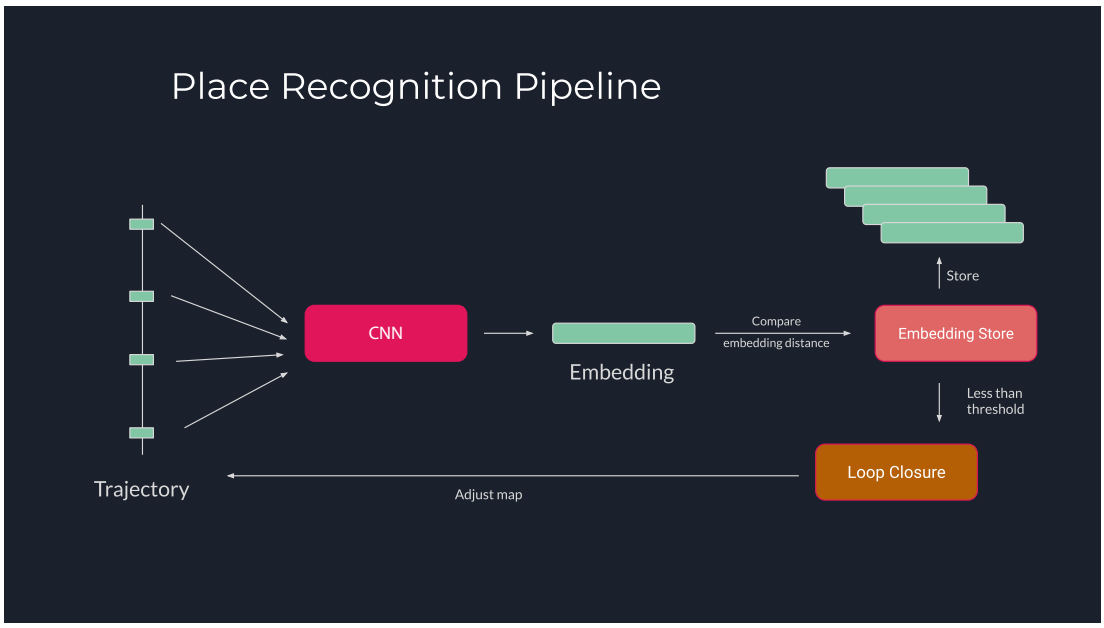


Figure 4.3: Place Recognition pipeline

at different layers. The network is trained to optimise the embeddings such that the embeddings of images of the same locations are similar, i.e., closer in the *embedding space*, and the embeddings of images with different labels are well separated in the embedding space. Typically, a weakly supervised learning paradigm can be used in such a setup where pairs of positive images and negative images (or triplets containing an anchor, a positive and a negative) are utilised to train with contrastive losses. After training, the VPR is deployed within a SLAM system, where we infer an embedding for each new keyframe and compute the distance to the existing embeddings. If there are distances less than a predefined threshold, we predict that the robot platform may have reached a place it has previously seen, adding the corresponding images to the loop closure candidates list. Figure 4.3 illustrates the pipeline using an embedding-generating CNN. We now detail in this section various methodologies used, starting with the dataset used.

In weakly supervised learning, two major loss functions used are pairwise contrastive loss (also known as pairwise ranking loss or simply contrastive loss) and triplet loss.

Pairwise contrastive loss penalises distant positive pairs and negative pairs closer than a margin. In the embedding space, this encourages the network to map positive pairs closer together while pushing negative pairs further apart. It is given by,

$$J(I_x, I_{x^*}, y) = yd^2 + (1-y)\max(m - d^2, 0) \tag{4.1}$$

where,

J = the loss function
I_x = anchor image
I_{x^*} = sampled matching image
y = 1 if x and x^* have same label, 0 otherwise
d = the distance between the embeddings computed for I_x and I_{x^*}
m = margin

During inference, the prediction $\hat{y}$ is computed as 1 (indicating a match) when $d \leq \gamma$, and 0 otherwise, where γ is the matching threshold. The matching threshold γ is initially set to half the margin during the training phase, and then the Receiver Operating Characteristic (ROC) curve is used to finetune the threshold γ for inference by choosing the right trade-off between precision and recall.

In a fully trained pairwise contrastive loss model, all points with the same label should be coincident with each other in the embedding space in order to make the loss contributed by $d(x, x^+)$ 0. However, in many cases, forcing the embeddings of the same class to collapse to a point may result in a degenerate behaviour whereby the network begins mapping negative samples closer to the positives.

Triplet loss [63] ameliorates this issue by being less *greedy* than the pairwise contrastive loss. Triplet loss [195, 63] is given by,

$$J(I_x, I_{x^+}, I_{x^-}) = [\, d(x, x^+) + m - d(x, x^-) \,]_+ \tag{4.2}$$

where,

J = the loss function
I_x = anchor image
I_{x^+} = image with same label as anchor
I_{x^-} = image with label different from that of the anchor
x = embedding for I_x
$d(p, q)$ = the distance between the embeddings p and q
m = margin
$[\text{value}]_+$ = max(value, 0)

In a fully trained triplet loss model, all points with the same label are allowed to have a non-zero intra-class separation in the embedding space while still being separated from the other classes by at least margin m. Triplet loss minimises the difference between $d(x, x^+)$ and $d(x, x^-)$, i.e., positive values for $d(x, x^+)$ do not count towards the loss as long as they are greater than $d(x, x^-)$, unlike pairwise contrastive loss that tries to bring the absolute distances $d(x, x^+)$ and $d(x, x^-)$ separately to zero and greater than margin, respectively.

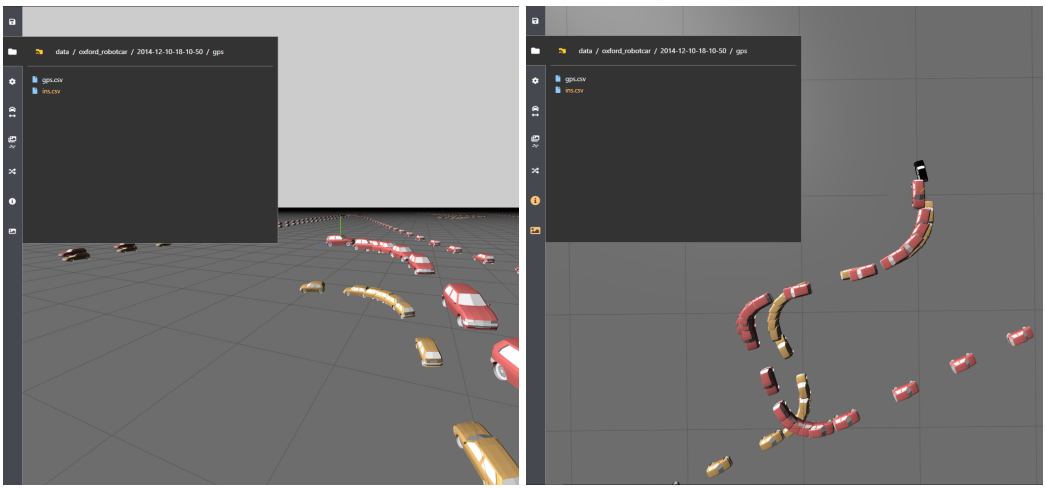
Embedding similarity can be measured using various norms. The most commonly used are (i) the Euclidean distance, or (ii) the cosine similarity. The Euclidean distance corresponds to the straight-line distance between two points in the embedding space. It is calculated by $\sqrt{\sum_{1..n}^{n}(q_i - p_i)^2}$, where p and q are two embeddings with n dimensions and p_i and q_i are their coefficients for i^{th} dimension. The cosine similarity measures the cosine of the angle between two non-zero vectors, calculated by $\cos\theta = \frac{p \cdot q}{\|p\|\|q\|}$, where $\|x\|$ is the norm of the vector x.

In our experiments, we use triplet loss with embedding distance calculated using Euclidean distance. In this section, we detail the methodologies employed to train robust learned descriptors using triplet loss, beginning with the process of data curation from large sequential datasets.

4.3.1 Curating training data

We use the Oxford RobotCar dataset [7], a large public sequential dataset exhibiting extensive seasonal and time-of-day variances for our experiments. We derive training data from it to train the VPR model to operate under challenging conditions. In the dataset, data from the GPS, the INS, and the camera are not synchronised with image data, as they operate at different frequencies: 5 Hz, 50 Hz and 16 Hz respectively. We obtain the pose for each image with the closest fused GPS+INS reading (continuous corrected GPS data with integrated inertial measurements). We note that interpolation of the INS was not required given the much higher sampling rate of the INS (50 Hz) when compared to the camera (16 Hz).

We note that we also make use of another dataset, Freiburg Across Seasons [128], for a small subset of experiments to make certain key decisions, such as choosing the



(a) Perspective View (b) Top View

Figure 4.4: Images showing the inaccuracies in the altitude recorded by GPS. Poses with different timestamps are differentiated using a colour transition from red to yellow from start to end. Perspective view shows the difference in altitude between the same start and end location. In contrast, Top View shows that the latitudes and longitudes remain closely overlapped for these poses.

backbone and ensuring training feasibility, prior to conducting the main experiments.

GPS data recorded also contains altitude[2] measurements. However, altitude measured by GPS is not reliable enough to supersede the barometric altimeter readings [177]. The general rule of thumb is that vertical error in GPS readings is up to three times the horizontal error, and it is not uncommon for satellite-derived heights to differ from map elevations by ±120 m [178]. We observed an altitude mismatch in the logs of a randomly taken traversal of the Oxford RobotCar main trajectory was about 10 m between the start and endpoint; see Figure 4.4. Therefore, we exclude altitude measurements when calculating distances during the matching process, as previously described in Section 3.4.2.1. As such, caution was exercised when using data from regions that included roads at different altitudes. We also carried out a visual check using OdoViz [24] to ensure that there were no incorrect matches due to the collapse of altitudes. We calculate GPS distances using the Haversine Formula[3], as it works reliably for finding small geodesic distances [182]. We obtain dense correspondences for each frame in the main trajectory using weighted distance and heading difference, maximising the visual content overlap.

[2]GPS receivers can also determine altitude by trilateration with four or more satellites

[3]The Haversine Formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes

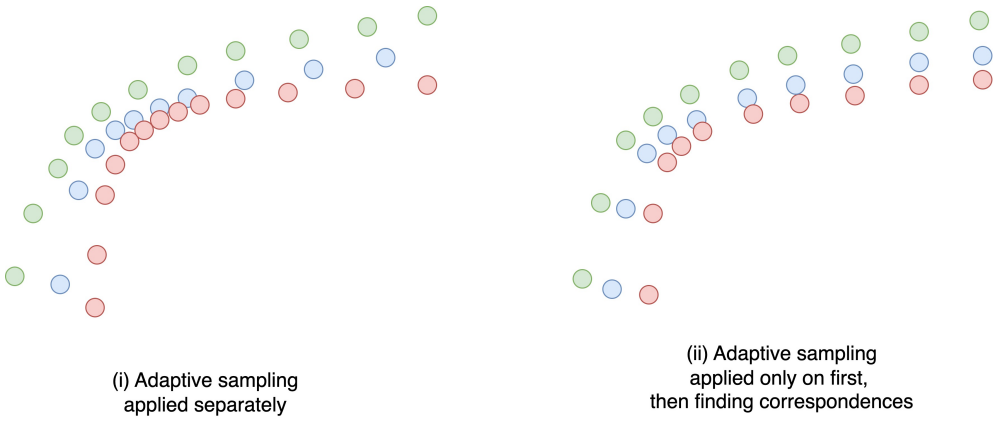


Figure 4.5: Illustration showing top view of poses from three traversals of the same trajectory. Left: Adaptive sampling is applied independently to each of the three traversals, resulting in distinct sampled poses for each trajectory, which can lead to challenges in establishing correspondences across the traversals. Right: Adaptive sampling is applied only to a single reference traversal (shown in green), and the corresponding poses in the other traversals are matched to the sampled poses of the reference traversal.

Situations where the car stops or waits at a traffic signal can result in hundreds of images with nearly identical latitude and longitude but with negligible or no visual content differences. For example, if the car waits for 20 seconds and the camera captures at 30 fps, this results in 600 images with negligible changes in GPS values, bearing minimal viewpoint variation. Using all images from a chosen trajectory as anchors will thus cause a data bias in the neural network, which will impede the learning process. For a single traversal, this can be effectively resolved by using adaptive sampling from OdoViz [24], as we previously explained in Section 3.4. However, repeating the same process on the poses from each traversal of the same trajectory will result in decimating poses closer to anchor images in other traversals; see Figure 4.5. We address this issue by applying adaptive sampling exclusively to a reference traversal and subsequently finding correspondences for each of the filtered poses within this traversal.

4.3.2 Discretising trajectories into locations

Weakly supervised training data for the VPR model requires sampling several thousand data records, each containing an anchor image, a matching positive image, and a non-

matching negative image. It is important that the triplets (or pairs) in the training data are not pre-computed prior to training, as training on the same fixed pairings for multiple epochs would hinder learning. As such, dynamic pairing of positives and negatives is essential for more generalised learning.

One common method employed in existing research works is to set a static GPS distance threshold to obtain ground truth positives for each anchor image [66, 67]. Among these positives, the image closest to the anchor image is chosen as the positive image. Alternatively, the positive match can be sampled from images in the same region that lie within a specified static GPS distance threshold [66, 63]. The negative image is then sampled from a list of negatives whose descriptors are closer to the anchor than the selected positive [67]. However, such methods using a static GPS distance threshold result in (i) false positives: large static thresholds over 25 m produce false matches that do not share any features, particularly when significant changes in heading are involved; (ii) false negatives: a small threshold, less than 10 m will result in images with common features along long straight roads being marked as negatives. Furthermore, sampling positive poses within the distance threshold but facing opposite directions (i.e., with 180° a difference in heading) can result in images that do not share any common visual features.

To this end, we propose discretising trajectories into locations or regions that contain similar images with consistent visual appearance. This discretisation enables the positive images to be sampled from within the same location as that of the anchor image and the negative images to be sampled from any of the other locations. We consider one epoch to be complete after one anchor image is drawn from all such locations. This setting allows the triplet combinations to be unique each time for each anchor image sampled from discrete locations, while also avoiding training images excessively from a single confined region (due to image capture rate and stationary vehicle).

Subsequent to dense correspondence matching, we use both an accumulated distance threshold d_{acc} and an accumulated heading difference threshold θ_{acc} to discretise trajectories into locations. This approach is thus well-suited to the problem at hand, mitigating the previously mentioned issues (i) and (ii), including the challenge of selecting positive samples from a wider region around corners where minimal visual content overlap occurs due to the larger angular variation. From the top view of the trajectory, the locations can be visualised as circles with a maximum radius of d_{acc} each composed

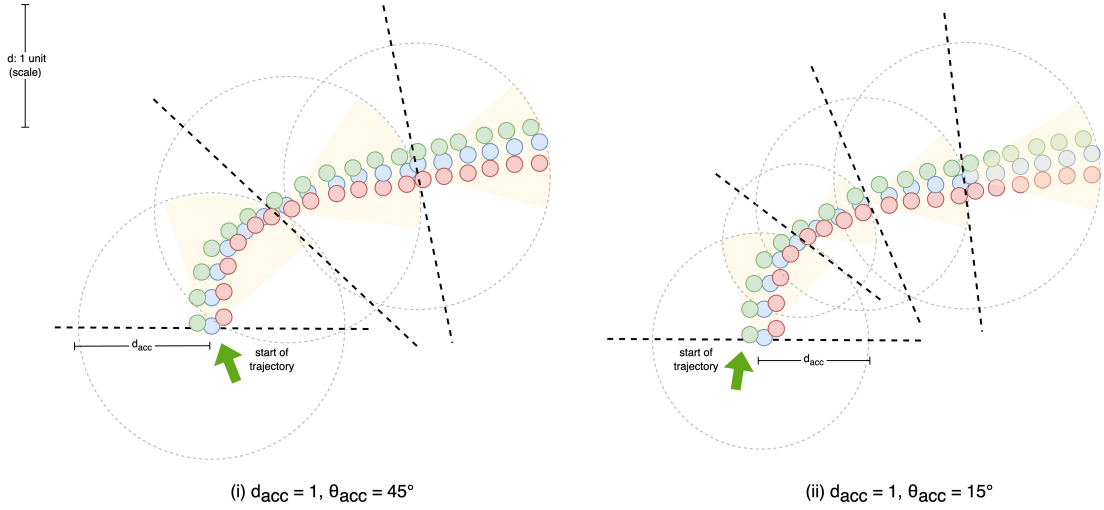


Figure 4.6: Illustration showing the top view of trajectory poses in 2D with locations as circles with a maximum radius determined by the accumulated distance d_{acc}. The poses within each location can be seen concentrated mostly within a sector (shaded pale yellow) constrained by an accumulated angle cut-off threshold θ_{acc}. Note that when $\theta_{\text{acc}} = 45°$ (left) results in d_{acc} takes precedence, resulting in uniformly distributed and equally sized locations. Conversely, a smaller value $\theta_{\text{acc}} = 15°$ (right) leads to the creation of more locations, with smaller-sized locations appearing in corners and larger-sized locations forming along straight roads. Tilted dashed lines along the diameter of the circle are included to aid in perceiving the circle's centre and radius more clearly. Best viewed in colour, zoomed on a computer screen.

of multiple poses. However, the poses are concentrated predominantly within a sector of the circle, constrained by an angle θ_{acc}; see Figure 4.6. Specifically, when poses can no longer be contained within this sector with the angle constraint, a new location is generated to accommodate the subsequent poses. We highlight the importance of employing accumulated distances and heading differences rather than relying solely on instantaneous distances and heading differences. This approach is crucial because multiple small heading differences between consecutive poses can collectively exceed the angular bounds of a fixed sector, even if each individual difference remains below a predefined threshold. Similarly, the accumulated distance threshold ensures that discretisation occurs with bounded intervals (i.e., the diameter of the location has an upper bound d_{acc}), regardless of variations in vehicle speed. In contrast, solely measuring individual distances could lead to unbounded discretisation, particularly when the vehicle's speed fluctuates, thus affecting the precision of location tracking.

In our experiments, we set $\theta_{\text{acc}} = 15°$ and $d_{\text{acc}} = 1\text{m}$. We note that this discretisation also allows retrieving positive matches on the fly (detailed later in Section 4.3.4) eschewing the need to randomise, match, and pre-build for several tens of thousands of anchor images obtained directly from the public dataset.

4.3.3 Location Aggregation and Image Augmentation

Although such a discretised setting is amenable to learning invariant representations of the same scene across different times of the day and different seasons, sampling positive images only within the corresponding location is often insufficient to produce necessary viewpoint variance in the training data. To facilitate this, we group g_+ consecutive locations, allowing for positives to be sampled from a wider region, while limiting negative matches to be outside g_- nearby aggregated locations to prevent false negatives. Figure 4.7 explains using illustrations the effect of discretisation and using the aforementioned constraints.

We use $g_+ = 10$ and $g_- = 5$ in our experiments. With an accumulated distance $d_{\text{acc}} = 1\,\text{m}$ (and assuming no angular constraint), this enables positives to be sampled within up to 10 m for viewpoint variance, and negatives from at least 50 m away to avoid any overlap. Optimal settings may vary depending on d_{acc}, θ_{acc}, and dataset attributes like vehicle speed, camera capture frequency, and road geometry.

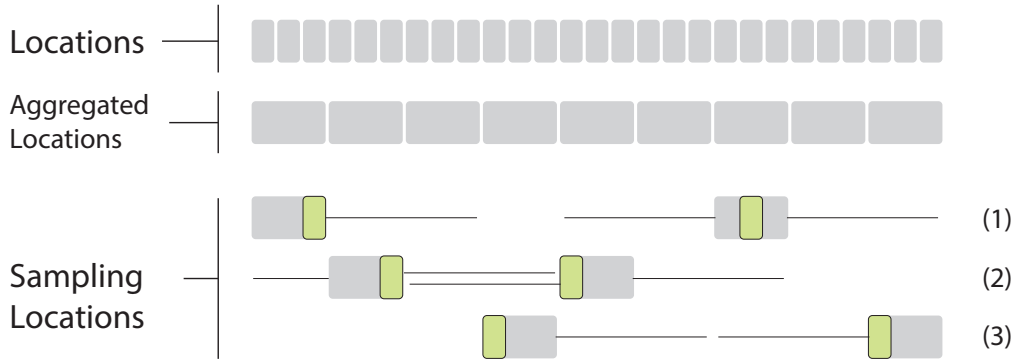


Figure 4.7: Illustration showing aggregation of nearby locations and 3 instances of sampling in the last row where positives are sampled with $g_+ = 3$ and negatives with $g_- = 2$. Big grey boxes indicate aggregated locations for sampling positives, and grey restriction lines on either side of grey boxes indicate locations from which negatives should not be sampled.

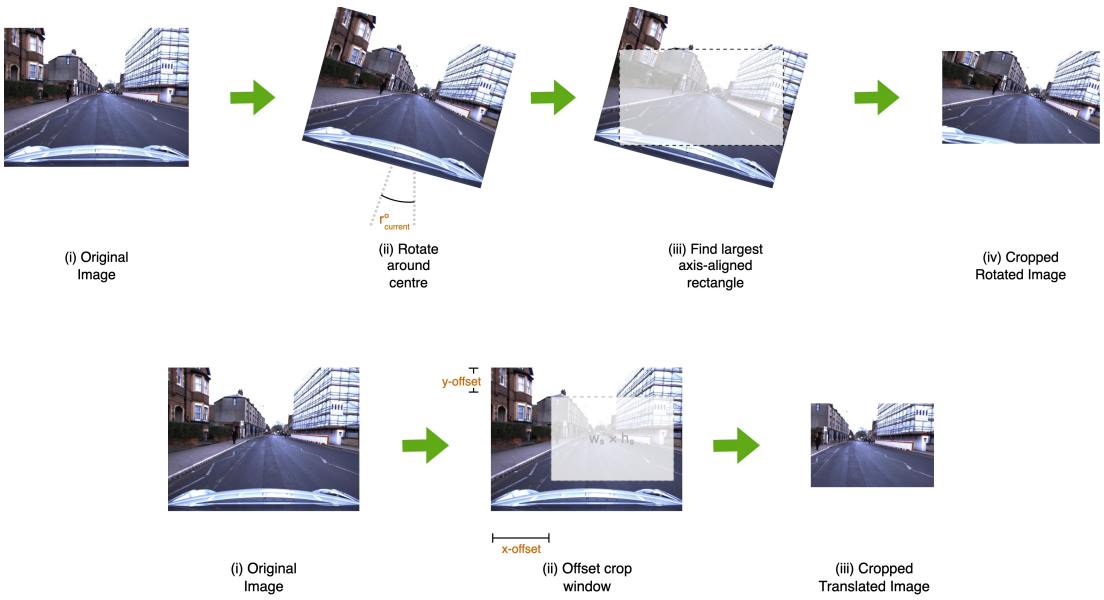


Figure 4.8: Illustration showing reformulated aspect-ratio preserving image augmentation operations, with rotations shown in the top row and translations in the bottom row. For rotation, the angle r_{current} is randomly sampled from a uniform distribution within the range [-r, r]. For translation, the offsets x_{offset} and y_{offset} are randomly selected, constrained by the maximum allowable values determined by the dimensions of the crop window $w_s \times h_s$. The image used is sourced from the *2014-12-09-13-21-02* trajectory of the Oxford RobotCar dataset [7].

To further promote viewpoint variance, we augmented our data with a series of bounded randomised (randomised for each image, every epoch) aspect-ratio preserving affine transformations where the images were rotated, translated, zoomed, and cropped. It should be noted that when performing these transformations, we do not fill the resultant excess pixels using constant, same, reflect, or wrap modes [196]. Such transformations result in synthesising image regions that do not imitate what we would encounter in the real world, diverging from VPR objectives. To this end, we uniformly sample from $[-r, r]$ to apply rotations about the centre, r being the maximum rotation angle. Then we clip the image to the largest axis-aligned rectangle (i.e., rectangle with maximal area) within the rotated image. Furthermore, we formulate the translation operation as an aspect-preserving crop operation to prevent translation from cropping out of the image. In more detail, we first choose a crop sub-window of size $[w_s, h_s]$, and then we individually uniformly sample the translation offsets for the crop window. Figure 4.8 illustrates the steps for reformulated rotation and translation augmentation operations.

Additionally, we introduce an intermediate resize step where the images from the dataset are initially reshaped to a larger resolution (e.g., 960×540) than what is originally required before image augmentation operations. This is to avoid blurred images as a result of performing reformulated image operations on the otherwise fully cropped smaller resolution (e.g., 240×135) images. It should be noted that we use a wider $16:9$ aspect ratio instead of a standard $1:1$ adopted in many popular works [57, 58] for other image tasks, as the visual content on either side of the road present in the left and right sides of the images often contains important visual cues necessary for place recognition. We publish our implementation of the aforementioned reformulated image augmentation operations as a public PyPI package[4] facilitating both reproducibility and reuse by the research community.

4.3.4 Building Batches Online and Batch Loss Strategy

Triplet loss requires processing three images at once, which in turn requires a network with three branches with shared weights, one for inferring each image in a triplet. Figure 4.9 shows the standard triplet network architecture. Training data for such a

[4]Code open-sourced as a Python *pip* package at PyPI https://pypi.org/project/imaugtools/

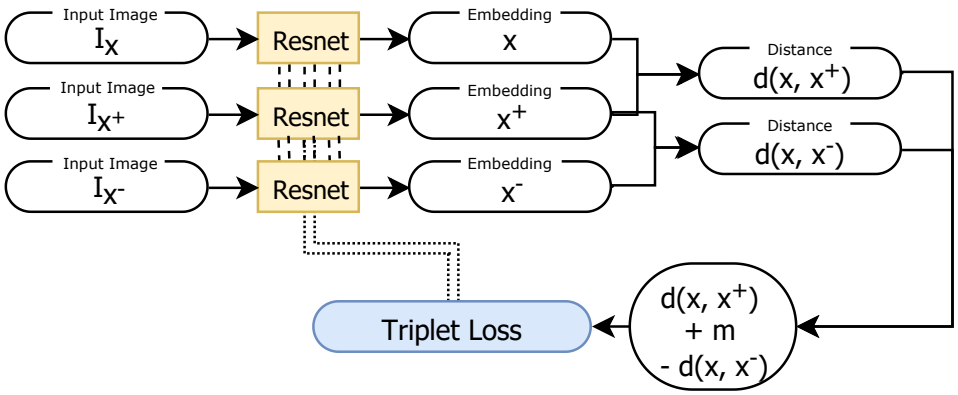


Figure 4.9: Triplet Network Architecture showing a triplet (i.e., one record of data comprising an anchor image and its positive and negative images) being processed by three branches with shared weights. The distance of the anchor to its positive embedding $d(x, x^+)$ and that of its negative embedding $d(x, x^-)$ along with margin m is then used to obtain the triplet loss, which is then used to adjust the weights of the shared ResNet backbone.

network requires building batches of triplets curated and resampled before every epoch so that the network is trained with different combinations of positives and negatives for each anchor image for better generalisability of the network.

Although the discretisation alleviates the process of finding matching positive samples at random, the time-consuming task of pre-generating combinations prior to each training epoch remains an issue. To deal with this issue, we build triplets in an online fashion by carefully selecting aggregated locations from which images will be sampled. More specifically, the triplets are constructed after the images are mapped to embeddings and before calculating the loss, rather than building all triplets needed for an entire epoch, as described earlier.

First, we assign indices for aggregated locations and treat them as classes (which we refer to as location classes), akin to classes in face recognition datasets. For each batch, we randomly choose L location classes that are at least g_- apart from one another and sample $n_s(L)$ images from each class. Next, during training, for each batch, inference is carried out up-front on the GPU for all images in the batch, following which a pairwise distance matrix is computed, i.e., distances from and to all embeddings. We then make triplets utilising the prior knowledge that $n_s(L)$ similar images have been sampled from

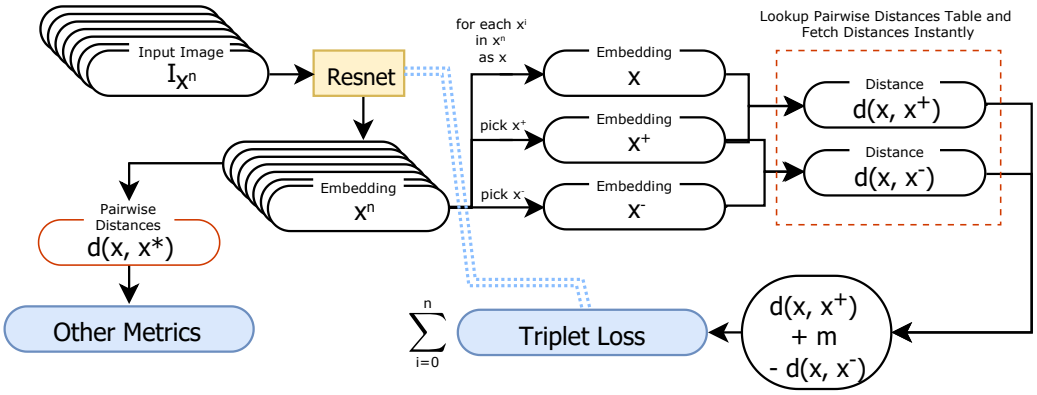


Figure 4.10: Modified architecture to process the batches built online by sampling $n_s(L)$ images from L location classes. All the input images $\{I_x\}$ are mapped to their embeddings $\{x\}$ using the backbone, followed by building a pairwise distance matrix. Then, the triplets are curated online based on the location class labels. The distances required are instantly fetched by looking up the pairwise distances matrix to compute the triplet loss efficiently, which is then backpropagated to update the backbone's weights. This setup eschews the complex process of creating triplets beforehand at the start of each epoch, saving a significant amount of compute and time.

L distinct locations. Thus, each image should have $n_s(L)-1$ positives and $(L-1)(n_s(L))$ negatives, resulting in a balanced batch where the number of valid triplets available for each anchor is the same as that of the other anchors. We refer the reader to Equation 4.3 and Equation 4.4 for an explanation with an example.

Distances required for the loss function, i.e., the distance between embeddings of each anchor to its assigned positive and negative samples, can then be obtained efficiently from the pre-computed pairwise distance matrix. Figure 4.10 shows the network architecture using online triplet loss, where the triplet pairings are computed from within the batch, followed by computing the loss utilising the pre-computed distance values for backpropagation.

Thus, when building batches online, triplets are constructed efficiently with less compute, distributed amongst the batches. In contrast, with offline batch building, triplets are computed for all anchors in the epoch, where computing possible triplets for each anchor would require substantial memory, making it impractical for large-scale datasets.

We provide an example for a better understanding of this concept and to further explain the loss strategies. Assume we have a dataset consisting of 5 traversals of a trajectory discretised into 1000 locations, aggregated with $n_{\text{agg}} = 100$ to form 100 location classes. Note that each location class contains 5 images ideally, but if a match is not found, i.e., when no candidate pose is within the dynamic distance threshold for matching, we will have fewer than 5 images. Thus, we would sample less than 5 from each location to avoid imbalance in the batch. If we randomly sample 10 locations (L) with a constraint of $g_{-} = 5$, each with $n_s(L) = 3$ randomly sampled images, then,

$$\begin{aligned} \text{batch size, } b &= L \times n_s(L) \\ b &= 10 \times 3 = 30 \end{aligned} \tag{4.3}$$

where, $n_s(L)$ denotes the number of images sampled from L.

We have 30 images in total, and each image acts as an anchor. For each anchor I_{x_i}, we have 2 positive images $I_{x_i^+}$ that belong to the same location, with the remaining 27 consisting of negative images $I_{x_i^-}$ that belong to different locations. These numbers are calculated as follows:

$$\begin{aligned} n(I_{x_i^+}) &= n_s(L) - 1 \\ &= 3 - 1 = 2 \end{aligned} \tag{4.4}$$

$$\begin{aligned} n(I_{x_i^-}) &= (L-1) \times n_s(L) \\ &= (10-1) \times 3 = 27 \end{aligned} \tag{4.5}$$

Hence, for each anchor I_{x_i}, we can have 54 valid triplets of the form $(I_{x_i}, I_{x_i^+}, I_{x_i^-})$, given by,

$$\begin{aligned} n(I_{x_i}) &= n(I_{x_i^+}) \times n(I_{x_i^-}) \\ &= 2 \times 27 = 54 \end{aligned} \tag{4.6}$$

where, $n(I_{x_i})$ is the number of valid triplets for I_{x_i}.

Batch All: In *Batch All* strategy [63, 65], we backpropagate the loss signal of all valid triplets for each anchor within the batch. If we have b anchors in a batch, we construct a total of $b \times n_t(I_{x_i})$ triplets, picking all possible triplets for each anchor.

Batch All loss is given by,

$$J(\text{batch}) = \sum_{i=1}^{b} \left(\sum_{\forall \left(I_{x_i}, I_{x_i^+}, I_{x_i^-} \right) \in B} J(I_{x_i}, I_{x_i^+}, I_{x_i^-}) \right) \tag{4.7}$$

where,

$J(\text{batch})$ = Total loss for the batch B
B = Batch of b images: $I_{x_0}, I_{x_1}, I_{x_2}, ..., I_{x_b}$
J = Triplet loss as explained in Section 4.3
$I_{x_i^+}$ = image whose x_i^+ is the valid positive for anchor I_{x_i} within the batch
$I_{x_i^+}$ = image whose x_i^- is the valid negative for anchor I_{x_i} within the batch

Batch Hard: In *Batch Hard* strategy [63, 65], we backpropagate the loss signal of only the hardest triplets for each anchor within the batch. If we have b anchors in a batch, we construct a total of b triplets, selecting the hardest triplet for each anchor. The hardest triplet is composed of the positive image with the largest embedding distance and the negative image with the smallest embedding distance.

Batch Hard Loss is given by,

$$J(\text{batch}) = \sum_{i=1}^{b} J(I_{x_i}, I_{x_i^{\#+}}, I_{x_i^{\#-}}) \tag{4.8}$$

where,

$J(\text{batch})$ = Total Loss for the batch of b images: $I_{x_0}, I_{x_1}, I_{x_2}, ..., I_{x_b}$
J = Triplet loss as explained in Section 4.3
$I_{x_i^{\#+}}$ = image whose $x_i^{\#+}$ is the hardest positive for anchor I_{x_i} within the batch
$I_{x_i^{\#+}}$ = image whose $x_i^{\#-}$ is the hardest negative for anchor I_{x_i} within the batch

In theory, *Batch All* strategy should be more suited for networks that require training from scratch since it backpropagates the error signal for all triplets. A significant limitation of this strategy is that the number of possible triplets grows cubically with the size of the dataset, making training increasingly impractical for large datasets. Compounding this issue, the model quickly learns to map most trivial triplets correctly, leaving a substantial portion of the triplets uninformative. Thus, mining hard triplets becomes crucial for learning [65].

Notably, unlike selecting the hardest samples from the entire dataset — which can lead to stalled learning or training failure due to overly difficult examples — *Batch Hard* strategy utilises the hardest positives and negatives within each mini-batch. This ensures that challenging examples are used to drive learning while avoiding the risk of only selecting excessively difficult cases or outliers that could hinder convergence.

While *Batch Hard* strategy might still work for training from scratch, we found that in our experiments, it often resulted in training failures, as explained in detail in Section 4.3.5. It necessitated careful hyperparameter tuning, requiring more manual effort, compute, and time. Moreover, the results may not be transferable to another dataset or a modified configuration. To avoid these issues, we chose to build a custom loss function instead that does not require such careful tuning.

4.3.5 Custom Loss and Embedding Normalisation

During training, we found that the training phase can stall in one of the following ways:

1. **Embedding Collapse**: Embedding norm approaches 0.
2. **Embedding Explosion**: Embedding norm approaches infinity.
3. **Hyper-dense Embedding Space**: All the images get mapped to a very small region in the embedding space, and this region slowly collapses to a point, while the embedding norm itself is non-zero.
4. **Hyper-sparse Embedding Space**: All the images get mapped far apart in the embedding space, and the distance between embeddings approaches infinity.

We conducted an experiment on a small subset of the training data, focussing on variations in embedding space sparsity. To this end, we visually inspect and analyse changes in the distance gap between positive and negative pairs using histograms. Figure 4.11 illustrates the evolution of this distance gap as training progresses. It can be observed that, in the absence of normalisation, the distance gap between positive and negative pairs gradually diminishes over time. This behaviour is caused by the backbone mapping images to a small, constrained region in the embedding space as an adjustment to minimise the high loss associated with the distance between positive

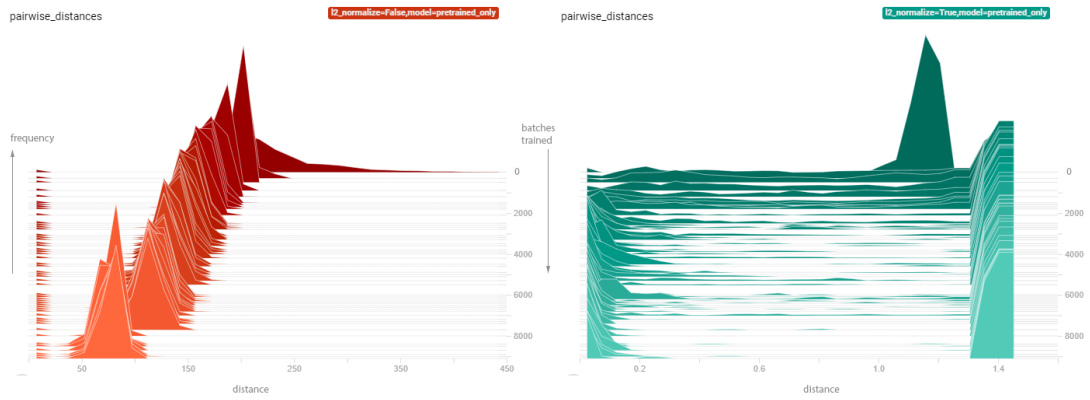


Figure 4.11: Histogram of batch pairwise distances matrix (distance to all embeddings, for each embedding in a batch) for different training batches stacked in the z-axis (most recent training batch to the front) shows the change in the distance values during training without L_2 Norm (left) and with L_2 Norm (right). During training, the separation of positive from negative samples is slower with L_2 Norm (right), while without L_2 Norm (left), the separation is quicker, but eventually the gap shrinks, leading to embedding space collapse and ultimately leading to training failure.

pairs. Such an overly compact embedding space significantly reduces the discriminative power, severely limiting the model's ability to separate positive and negative images effectively. We hypothesise that continued training in such cases leads to a *Hyper-dense Embedding Space*, where all images are indistinguishably mapped to a single point in the embedding space, effectively collapsing the representation.

When the embeddings are L_2 normalised, the embedding space is restricted to an n-dimensional hollow hypersphere where the embeddings are present only on the surface of the sphere, maintaining a constant vector norm. Thus, the introduction of L_2 norm addresses *Embedding Collapse* and *Embedding Explosion* issues. We also note that in such a setting, the Euclidean distance, cosine distance, and geodesic distance are all positively correlated and monotonically increasing. Therefore, it will not be particularly advantageous to use one distance metric over the other, as the effect would be equivalent to scaling the loss.

However, even with embedding normalisation enforced, the embedding space can still become overly compact, leading to a *Hyper-dense Embedding Space*, mapping all images to a single point in the embedding space. Conversely, consistent larger losses on the distance between negative pairs and/or a large margin value force the mappings to

be distant from one another, eventually leading to a *Hyper-sparse Embedding Space*.

To this end, we created a custom loss function modifying batch hard triplet loss that not only remains stable but also results in faster learning. In more detail, we introduce two more terms, $J_+(I_x)$ and $\frac{1}{J_-(I_x)}$, weighted with α and β respectively, to the underlying triplet loss. These terms control the distribution of positive and negative matches in the embedding space, respectively. This customisation takes advantage of the batch hard strategy while also preserving the embedding stability during training.

The new loss function is given by,

$$\begin{aligned} J^*(\text{batch}) &= \sum_{i=0}^{b} \left(J(I_{x_i}, I_{x_i^{\#+}}, I_{x_i^{\#-}}) + \alpha J_+(I_{x_i}) + \beta \frac{1}{J_-(I_{x_i})} \right) \\ J_+(I_x) &= \sum_{\forall I_x^+ \in B} d(x, x^+) \\ J_-(I_x) &= \sum_{\forall I_x^- \in B} d(x, x^-) \end{aligned} \tag{4.9}$$

where,

$J^*(\text{batch})$ = Total Loss for the batch B

J = Batch Hard Triplet loss as explained in Section 4.3

B = Batch of b images: $I_{x_0}, I_{x_1}, I_{x_2}, ..., I_{x_b}$

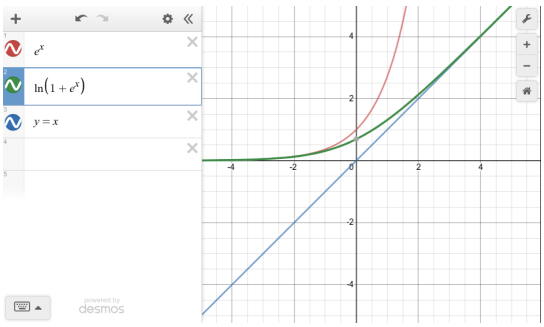
$I_{x_i^{\#+}}$ = image whose $x_i^{\#+}$ is the hardest positive for anchor I_{x_i} within the batch

$I_{x_i^{\#+}}$ = image whose $x_i^{\#-}$ is the hardest negative for anchor I_{x_i} within the batch

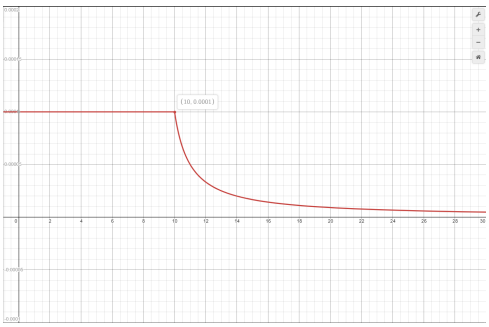
4.3.6 Architectural, Learning Rate and Other Adaptations

4.3.6.1 Convolutional Blocks

We add additional convolutional blocks at the end of the pretrained ResNetV2-50 architecture to allow learning further features specific to VPR. Each convolutional block consists of a ReLU-activated convolution layer with 3×3 filters, followed by a max-pooling block to downsize the feature maps by a factor of 2, following the standard convolutional block design used in [57].



(a) Softplus function shown in Bold Green, against Exp function (Red), and Identity Function (Blue)



(b) Graph showing exponentially decaying learning rate (y-axis) as number of training steps (x-axis, in thousands) increases

Figure 4.12: Graphs showing softplus function and exponential learning rate decay

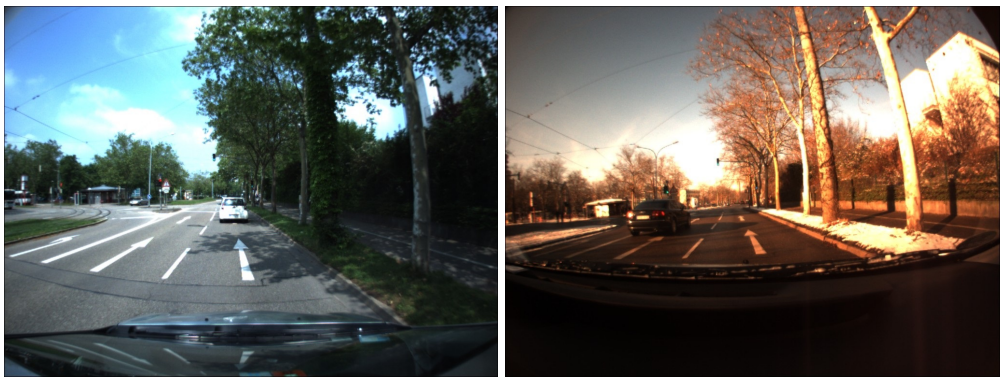
$d(x,x^+)-d(x,x^-)$	0.00	1.00	2.00	3.00	4.00	5.00	6.00	7.00
Soft Margin	0.69	0.31	0.12	0.04	0.01	0.00	0.00	0.00
Total	0.69	1.31	2.12	3.04	4.01	5.00	6.00	7.00

Table 4.1: Table showing how using softplus function changes the margin and total for different values of $d(x,x^+)-d(x,x^-)$, the difference between positive pair and negative pair distances

4.3.6.2 Soft Margin

Using a larger margin will stagger the training process and will lead to *Hyper-sparse Embedding Space* (explained in Section 4.3.5). This can also lead to instability even when using normalisation.

Using a softplus function, we can assign a variable margin based on the difference in positive pair and negative pair distances $d(x,x^+)-d(x,x^-)$, i.e., if the difference in distances is close to 0, the margin is set to 0.69 (total 0.69); for 1, it is set to 0.31 (total 1.31); and for 2, it is set to 0.12 (total 2.12). As the distance increases, the margin becomes equal to the actual difference and approaches zero. Figure 4.12a shows the graph of the softplus function and Table 4.1 the exponential decrease of the soft margin added as the $d(x,x^+)-d(x,x^-)$ difference increases.



(a) Summer, May 2012 (b) Winter, Dec 2012

Figure 4.13: Matching image pair from the same location in the Freiburg dataset [128]

4.3.6.3 Exponentially Decaying Learning Rate

When trained for a very large number of epochs, the network starts to overtrain, which in turn leads to overfitting. In our setup, the learning rate is decayed exponentially, which results in greater stability during the later stages of training whilst at the same time does not affect the existing training pipeline [65]; see Figure 4.12b.

4.4 Experiments

4.4.1 Backbone

Initially, we conduct exploratory experiments employing contrastive loss (pairwise ranking loss) within a Siamese architecture with shared weights to train on a smaller sequence-aligned Freiburg Across Seasons dataset [128]. Figure 4.13 shows an example from the dataset where the corresponding images exhibit large appearance changes.

We use a Siamese configuration shown in Figure 4.14 with three different backbones: VGG-16 [96], ResNetV2 [58], and InceptionV3 [197], all pretrained on ImageNet [96], sharing weights between the two branches of the network. To get embeddings of the same size, we include the final dense layers for VGG-16 and use *global average pooled* outputs from *headless*[5] models for ResNetV2 and InceptionV3.

[5]without the final fully connected and classification layer

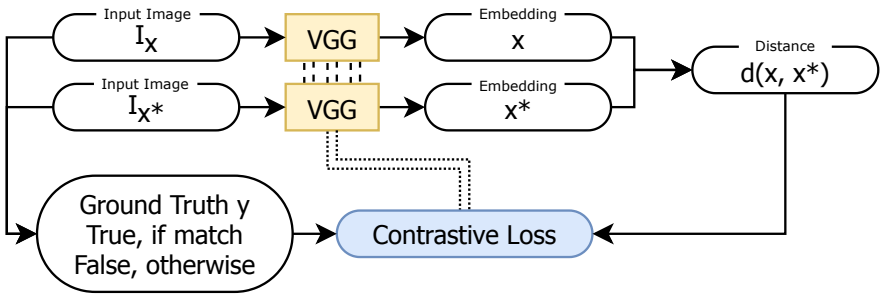


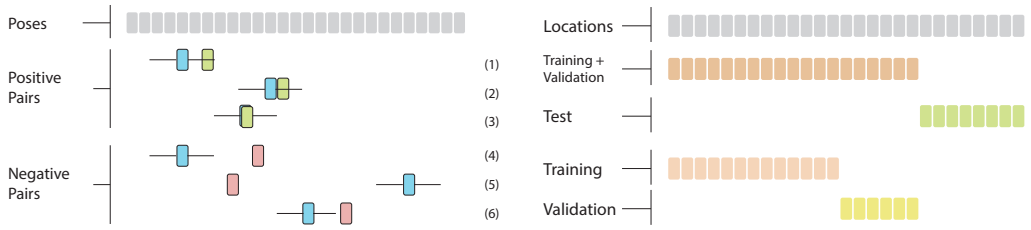
Figure 4.14: Siamese Network Architecture

From the dataset, we obtain 674 positive pairs from matching image correspondences across the two traversals (traversed in summer and winter). For each matching pair of images in the dataset, we generated a non-matching pair, picking a reference image at random, resulting in a total of 1348 image pairs. To further encourage added robustness to variations in viewpoint, we introduce constraints similar to g_+ and g_- used in Section 4.3.3. To cover a range of 10–15 m around the anchor pose, we randomly select positive matches from the 10 poses immediately preceding or succeeding it, in addition to using the matching image provided in the dataset. This introduces viewpoint variance among positive samples, based on a 1 Hz capture rate and typical vehicle speeds in the dataset. Similarly, to avoid sampled negatives being potential positives, we impose an additional constraint, where the non-matching image can be chosen at random but shall not be within the preceding or succeeding 10 frames of the actual reference image, as shown in Figure 4.15a. All images were then processed using the data augmentation pipeline we explained in Section 4.3.2.

We then train the network with an equal number of positive and negative pairs with weak supervision, i.e., metric learning. These experiments are intended to assist in selecting a suitable backbone for the subsequent sections of this chapter.

We reserve the last one-third of the trajectory for testing while using the rest for training. To gain a more comprehensive understanding of the results, we sliced the data twice with different regions as test data for a better understanding of the results:

- **Split 1**: The first one-third (450 image pairs) is used as test data for evaluation, and the remaining two-thirds of the data (898 image pairs) is used for training.



(a) Matching Restrictions (b) Data Split

Figure 4.15: (a) Illustration showing restriction bars (covering two preceding/succeeding frames) for query poses (blue bars) randomly sampled from sequentially arranged poses (grey bars) to show that positive matches (green bars) shall be chosen from poses covered by the restriction bar, while negative matches (red bars) are chosen outside it. (b) Illustration showing how data is split into training, validation, and testing batches. The split corresponds to Split 2 explained in Section 4.4.1

Backbone	Split 1		Split 2	
	AUC	AP	AUC	AP
VGG-16	0.9533	0.9534	0.6533	0.5768
ResNetV2	0.9681	0.9607	0.9668	0.9743
InceptionV3	0.9198	0.9354	0.5281	0.5814

Table 4.2: Experiment results showing Area Under the Curve (AUC) in the ROC graph and Average Precision (AP) for each model in a Siamese setting.

- **Split 2**: The last one-third is used as test data, and the first two-thirds is used as training data.

In both cases, the last 33% of the training data (300 of 898 image pairs) was used as validation data to monitor and improve the performance of the network. Figure 4.15b shows how data is split for training, validation, and evaluation phases.

Inference was performed on the test set using all three backbones of the network, and the Area Under the Curve (AUC) in the Receiver Operating Characteristic (RoC) curve and the Average Precision (AP) in the Precision-Recall (PR) curve were measured to evaluate their performance. Experimental results presented in Table 4.2 show the retrieval performance of the networks, VGG-16, ResNetV2, and InceptionV3. In Split

1, all the models can be seen to perform well with high AUCs of 0.95, 0.97 and 0.92 for VGG-16, ResNetV2 and InceptionV3 models, respectively, exhibiting their ability to learn the embedding space. However, in Split 2, VGG-16 and InceptionV3 models do *not* perform well, scoring AUCs of 0.65 and 0.53.

From this, we infer that ResNetV2 demonstrates greater robustness and generalisation capability across different data splits. In more detail, in Split 2, where the other models showed a significant drop in performance, ResNetV2 maintained its AUC and AP performance. Residual connections in ResNet architectures address the vanishing gradient problem commonly encountered in deeper neural networks. This capability enables more effective training and better feature extraction, allowing ResNetV2 to outperform VGG16 in computer vision tasks [58], which explains its superior performance. In the case of InceptionV3, we explored a range of hyperparameters and architectural configurations, including the inclusion of additional convolutional blocks after the base model, different pooling strategies, the use and rate of dropout, as well as training with and without pretrained weights. Despite this comprehensive tuning, InceptionV3 remained highly sensitive to these choices and did not consistently achieve AUC scores comparable to those of ResNetV2.

Given this, we pursue further experiments with a ResNetV2-50 backbone. It should be noted that in each of the experiments above, the dataset is limited in scale, and so the results only provide an indicative measure of each model's potential performance. As such, in order to comprehensively evaluate the proposed approach, a considerably larger dataset is required. In the next subsection, we utilise the Oxford RobotCar dataset [7], matching images across traversals to build a significantly larger training dataset exhibiting significant variability while avoiding repeated frames. We utilise the methodologies described in Section 4.3, and train the network on a ResNetV2-50 backbone using triplet loss.

4.4.2 Triplet Network

In this section, we further develop the idea of learning an embedding space using a ResNet-50 [58] backbone pretrained on ImageNet [96] with the Oxford RobotCar dataset [7], which provides a larger dataset containing over 100 traversals of a consistent route through Oxford exhibiting a wider variety of appearance changes: seasonal,

(a) 2014-12-09-13-21-02 (b) 2014-12-10-18-10-50 (c) 2015-05-19-14-06-38

Figure 4.16: Multiple views of the same location captured under different conditions: (a) Winter 2014 Day, (b) Winter 2014 Night, and (c) Summer 2015 Day

weather, and day-night illumination changes. For our experiments, we use 3 traversals from the Oxford RobotCar Dataset [7] main trajectory: *2014-12-09-13-21-02* (Daytime, Overcast, Winter), *2014-12-10-18-10-50* (Night, Winter), and *2015-05-19-14-06-38* (Daytime, Overcast, Summer), which demonstrate wide visual variability; see Figure 4.16 for samples.

Using the methodologies described earlier, we curate the necessary training data from the dataset by finding image correspondences, discretising trajectories into locations, and applying preprocessing steps. Next, we train the network equipped with architectural, loss function, normalisation, and learning rate adaptations, building batches online, as previously detailed in Section 4.3.1 to Section 4.3.6.

We use an augmentation probability p_{aug} of 0.5, wherein the images have a 50% probability of being augmented, thereby allowing an equal amount of original and augmented images to be present in the training data. The input to the network is set to a final resolution 240×135 after preprocessing. Similar to the train, validation, and test split used in Section 4.4.1, we allot one-third of the data for testing, and from the remaining two-thirds, we use 67% for training and the rest for validation. Additionally, we create an auxiliary validation set, using one-third of images sampled within each aggregated location. This secondary validation set aids in measuring the network's performance on new images from locations it has been trained on. Figure 4.17 illustrates how the data is split into training, validation, auxiliary validation, and test sets.

When integrated into a complete autonomous system, the model's predictions become helpful when the top matches are eliminated using additional information available. An example of such an approach is demonstrated in the work by Kim and

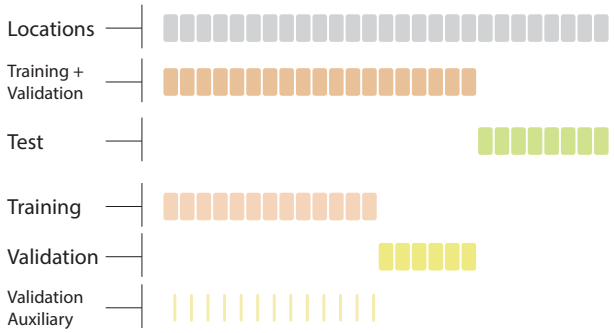


Figure 4.17: Illustration showing how data is split into training, validation, and testing batches. The auxiliary validation set is made by setting aside a small portion of images in each location reserved for training.

Walter [66], where they perform a Bayesian filtering of the output of their network through the use of a particle filter. Therefore, the presence of true matches in the top-k embeddings matches[6] is often more important in image retrieval than the actual separation of embeddings based on the threshold, and consequently, top-k retrieval metrics prove to be a useful performance indicator. Hence, in addition to deriving AUC (Area Under the Curve) in the ROC (Receiver Operating Characteristic) curve and AP (Average Precision) in the PR (Precision-Recall) curve, we also monitor precision@k and recall@k metrics to evaluate the model's performance. By noticing both precision@k and recall@k, appropriate decisions can be made, such as (i) introducing elimination mechanisms when recall@k is high and precision@k is low, and (ii) relaxing the thresholds when precision@k1 is high and precision@k2 is low, where $k1 < k2$. In addition to the metrics mentioned above, we additionally calculate the *Normalised Mutual Information* score (NMI Score, as explained in Section 3.5.5) for the embeddings inferred in a training batch to monitor the quality of clustering in the embedding space.

We present results on the performance of the trained network over the baseline on the full test set in Table 4.3, demonstrating the effectiveness of the proposed set of techniques.

[6] top-k results from batch images ordered by embedding similarity

Model	AUC	NMI	R@1	R@3	P@3
Baseline	0.7542	0.5165	0.3233	0.4767	0.2430
Improved (Ours)	0.8818	0.6166	0.5582	0.7518	0.4329

Table 4.3: Evaluation results demonstrate the overall improvement achieved over the baseline with various metrics. Both models utilise a ResNetV2-50 [58] backbone, with the improved model incorporating architectural modifications, a custom loss function, and other adaptations described in Section 4.3.4 - Section 4.3.6. Note that since precision@1 (P@1) and recall@1 (R@1) are equivalent (see Section 3.5.4), we present only a single column in the results table to avoid redundancy.

Model	Backbone	AUC
Random Initialised	ResNetV2-50	0.6058
ImageNet Pretrained	ResNetV2-50	0.6996

Table 4.4: AUC scores averaged over 3 training runs on validation data for randomly initialised and ImageNet [96] pretrained weights on the ResNetV2-50 backbone.

4.4.2.1 Ablation Studies

We conducted an ablation study on ResNetV2-50 to assess the impact of individual techniques on model performance. First, we initialised the weights of the headless ResNetV2-50 [58] backbone using two approaches: (i) random initialisation for training from scratch, and (ii) weights pretrained on ImageNet [96], originally optimised for classification tasks. The results of this comparison are presented in Table 4.4.

Additional experiments were conducted to assess whether appending a convolutional block to the pretrained backbone enhances performance. The appended block consisted of a 3x3 convolution layer with *same* padding, followed by ReLU activation, batch normalisation, max pooling, and a dropout layer with a rate of 0.25. The analysis included assessing the effect of adding this block to both a fully trainable pretrained network and a frozen pretrained network. Additionally, the influence of introducing an L_2-Normalisation layer to the architecture was examined. We present the results of this study in Table 4.5.

We observed that we get better performance with added convolutional blocks at the end at the expense of adding more (trainable) parameters to the neural network.

Model Configuration	AP	AUC	R@1	R@3	P@3
Without Normalisation					
frozen pretrained, cb	0.96	0.55	0.32	0.45	0.16
pretrained, cb	0.97	0.58	0.30	0.47	0.22
pretrained	0.96	0.58	0.35	0.40	0.19
With Normalisation					
frozen pretrained, cb	0.97	0.63	0.35	0.42	0.21
pretrained, cb	0.97	0.63	0.34	0.45	0.22
pretrained	0.97	0.56	0.19	0.26	0.09

Table 4.5: Table shows results on validation data after training 9000 batches, comparing with and without L_2 normalisation explained in Section 4.3.5. Three networks are used; all use ResNetV2-50 pretrained on ImageNet [96] denoted as pretrained; the ones that have the pretrained network with frozen weights are marked as frozen; the networks with additional convolutional blocks added at the end are denoted by *cb*.

Custom Loss (α, β)	AUC
(0.25, 0.25)	0.64461
(0.50, 0.50)	0.62294
(0.75, 0.75)	0.70584

Table 4.6: Table shows results on validation data for models with the custom loss function: 3 different values of alpha beta are used, and the AUC scores are tabulated. For (α, β) in the custom loss function, values (0.1, 0.1) and (1,1) were used and were found to overfit; the results of these are not included in the table.

We note that without L_2 normalisation, the model initially trained faster than its L_2 normalised counterpart (see Figure 4.11), however, it often resulted in training failure due to training instabilities, as previously explained in Section 4.3.5, requiring multiple training restarts.

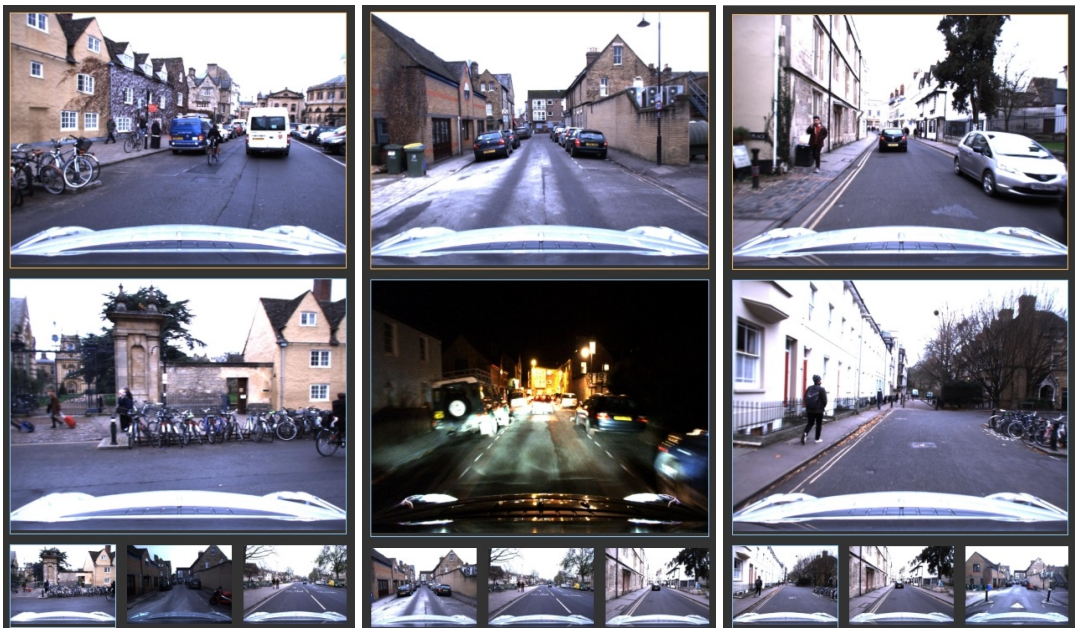
Additionally, we have also carried out experiments using the custom loss function with values [0.1, 0.25, 0.5, 0.75, 1] for both α and β where the results are tabulated in Table 4.6.

Based on the experimental results, we selected the optimal configuration to train the final model that includes a pretrained ResNetV2-50 backbone augmented with additional convolutional blocks, the inclusion of an L_2-Normalisation layer, and a custom loss function with $\alpha = 0.75$ and $\beta = 0.75$.

Additionally, we manually inspected the matches in the auxiliary validation set to understand if the network is able to match new images within previously seen locations correctly (see Figure 4.18) in addition to keeping track of metrics mentioned in the previous section. It can be seen from the figure that the retrieved images include day-night and cross-season image matches, supporting the validity of the embeddings. Manual inspection helped find biases, such as all top-k results being images from the same time of the day or season, etc. This was also very helpful in times when the network overfit training data, where it showed poor performance on the validation set but good performance on the training set.

4.5 Conclusion

In this chapter, we addressed the challenge of successfully training robust learned representations for VPR utilising data curated from a large sequential public dataset with weakly supervised learning. To do this, we employed our novel approach of discretising trajectories with sequential images into locations from which positive images can be sampled. We aggregated discretised locations and utilised data augmentation techniques tailored to obtain and use in training positive images with added viewpoint variance. We built training batches online during training, employing computational and memory optimisations by strategically building triplets from selective location classes after mapping images to embeddings, significantly saving compute time. We



(a) Correct Match (b) Correct Match (c) Incorrect Match

Figure 4.18: Retrieval results on the auxiliary validation set. (a) Anchor 80_2; Top-4 Retrievals: 80_5, 141_{16}, 141_{20}, 183_2 (b) Anchor: 141_{13}; Top-4 Retrievals: 141_{16}, 11_{10}, 70_{10}, 141_0, (c) Anchor 70_4; Top-4 Retrievals: 99_{25}, 70_{10}, 141_{11}, 116_{23}. In each image, Anchor (purple text) is on the top, Highlighted Retrieval (brown text) is in the middle, and the rest of the top 4 images are at the bottom. Notation L_i represents the i^{th} image in the location label L; any retrieval with L, same as that of the anchor is a correct match. We heavily rely on Top-k Image Retrieval Analysis Extension of OdoViz [24] for the interactive visualisation interface that allowed us to individually compare and analyse top-k retrieved images against the anchor.

utilised a custom loss function, adapting the architecture to successfully train on a large sequential dataset without incurring training failures. The proposed custom loss function and stabilisation techniques further enhanced the performance and reliability of our learned representations, addressing the challenges of visual place recognition in dynamic environments and challenging conditions.

Chapter 5

Scene Categorisation

5.1 Introduction

Figure 5.1: Images from the Scene Categorisation Dataset. Top Left: Rural (Utah), Top Right: Urban (Toronto), Bottom Left: Rural (Stockport to Buxton), Bottom Right: Suburban (Melbourne)

Scene categorisation is a fundamental challenge in computer vision that involves classifying an image into predefined categories, such as beach, restaurant, mall, and indoor areas, by understanding its overall content, objects, and their spatial layout. It requires reasoning about complex and diverse environments, aiming to provide contextual information about the scene, which is essential for intelligent systems to predict and interpret ongoing or future events. Unlike object recognition, scene categorisation faces challenges due to images from different classes often sharing similar objects, textures, and backgrounds, resulting in visual similarity and ambiguity among categories.

In the domain of autonomous navigation, scene categorisation offers a high-level description of the overall content of an image by classifying it into predefined categories, such as urban, suburban, or rural, without listing, identifying, or recognising individual objects in the scene. This approach is intended to assist mobile robots in gaining a more

holistic understanding of the surrounding environment. As such, scene categorisation is a precursor task with a broad range of applications in content-based image indexing and retrieval systems.

The retrieval accuracy of Content-Based Image Retrieval (CBIR) systems depends on both the feature representation and the similarity metric used. The retrieval process can be accelerated by selectively searching based on certain scene categories. For example, given a query image with multiple high-rise buildings, searching images from rural regions would not be beneficial and can be skipped. This knowledge about the scene can further assist in improving other computer vision tasks such as context-aware object detection, action recognition, and scene understanding [198, 122].

In autonomous driving scenarios, location context provides an important prior for parametrising autonomous behaviour. Generally, GPS data is used to determine if the vehicle has entered the city limits, where additional caution is required. This information is then used to appropriately adjust crucial thresholds for various tasks, e.g., to set the pedestrian detection thresholds and other perception hyperparameters. However, such an approach requires a priori labelling of the environment. Furthermore, due to the rapid development of regions around the cities and suburbs, it has become increasingly hard to distinguish such regions of interest based solely on GPS coordinates. A more scalable and lower-cost approach would be to automatically determine the scene type — urban, suburban, or rural — at the edge using locally sensed data.

Early deep learning classification approaches, such as those developed for image classification tasks on datasets like ImageNet [96] and PASCAL VOC [123], face limitations when directly applied to scene categorisation. In object-focussed tasks, these methods are designed to classify images where the view typically covers a range of 1 to 2 meters around the observer, concentrating on individual objects. However, in scene categorisation for autonomous driving scenarios, the *scene* encompasses a much larger area comprising several objects, typically extending beyond 5 meters from the observer. This broader spatial context is essential for understanding complex environments, making traditional object-focussed classifiers inadequate for such tasks [19].

Moreover, many learned image classification techniques often rely on end-to-end training that directly maps images to object classes without incorporating explicit intermediate representations in their architecture. Although techniques such as visualising

intermediate feature maps [199] and using Grad-CAM [200] or its variants can provide some insights into model behaviour, they do not offer structured intermediate representations that allow for a more detailed inspection of errors or failure cases. In contrast, models equipped with intermediate representations are known to train more efficiently, achieve higher task performance, and generalise better to previously unseen environments, providing greater transparency and robustness in their decision-making process [201].

Obtaining dense semantic segmentation labels, for example, using DeepLab [202] or RefineNet [86], provides more useful insights about the scene. However, obtaining the right scene category after obtaining semantic or geometric information is too computationally expensive for a precursor task. Also, as we discussed earlier in Chapter 4, the embeddings generated by such pretrained neural networks are heavily influenced by the training data. Retraining these networks with new data can be challenging, as acquiring human-annotated labels such as dense semantic segmentation labels is both labour-intensive and costly. To address this issue, in the previous chapter, we presented various techniques for training CNNs using weak supervision, avoiding training failures, and reducing the reliance on fully annotated datasets. In this chapter, to capture high-level scene features, we focus on unsupervised approaches that allow for scalability far beyond what weak supervision can achieve.

Human recognition of real-world scenes typically begins with the encoding of the overall scene configuration, with limited attention to finer details or individual object information. More specifically, human perception is generally not based on the initial identification of the objects within the scene [203]. Building on this insight, we utilise unsupervised learning approaches to be trained on vast amounts of data, such as hours of driving videos, without requiring manual or automated labelling of individual objects. Additionally, it mirrors the type of broad, adaptive learning humans engage in [204], offering greater flexibility and scalability in model development.

Within the domain of unsupervised learning of images, Variational Autoencoders (VAEs) [88] and Generative Adversarial Networks (GANs) [89] are two prominent approaches for learning data representations. VAEs have notable limitations, such as generating blurry images due to their reliance on Gaussian priors and their tendency to produce overly smooth outputs, making them unsuitable for producing high-resolution images.

Overcoming these issues, Generative Adversarial Networks (GANs) [89], quickly gained prominence by leveraging an adversarial training process between a generator and a discriminator. Subsequent works such as DCGAN [205], WGAN [206], and Progressive GANs [207] demonstrated significant advancements in generating realistic, high-resolution images, outperforming earlier generative models like VAEs in tasks such as image synthesis, super-resolution, and style transfer. However, the use of GANs for tasks beyond image generation remains limited, as their adversarial training process is primarily optimised for producing visually realistic outputs rather than learning robust, transferable feature representations suitable for tasks like classification, segmentation, or scene understanding.

More recent unsupervised-learnt image generation approaches, such as Vision Transformers (ViTs) [208] and Diffusion Models [209], demand significant computational resources, making them impractical for use as precursor tasks, especially in real-time applications where efficiency and speed are critical. We revisit the VAE architecture for use in scene categorisation and propose training it for the reconstruction task to capture scene features, utilising only the encoder to extract embeddings that are used as effective global feature descriptors.

Thus, the challenge lies in developing a model that is fast, efficient, and robust for scene categorisation in autonomous driving scenarios. The model must be capable of real-time performance, efficient enough to function as a precursor task without excessive computational overhead, and robust enough to handle diverse and changing landscapes. In this chapter, incorporating both unsupervised learning and intermediate representations, we:

- present a novel approach utilising a convolutional VAE to encode high-level scene information in a multi-dimensional latent space,
- train the VAE in an unsupervised fashion with image reconstruction as a proxy task for capturing high-level scene information without explicitly recognising objects, their semantics, or capturing finer details,
- propose to use disentangled latent variables as global feature descriptors and to serve as intermediate representations, allowing them to be used as more abstract and transferable feature representations,

- map these features to three scene categories: Rural, Urban and Suburban, using a light supervised classification head requiring less than 500 labelled images, and finally,
- present scene categorisation results where we demonstrate our technique to be fast and efficient with a compact embedding size of 128 and a compute time of 60 μs[1].

To summarise, this chapter presents a novel, efficient, and robust approach for unsupervised scene representation learning using convolutional VAEs. We detail the proposed methodology, describe the experimental setup, and evaluate our approach against established benchmarks. Furthermore, we provide a comprehensive comparison against benchmark learned and handcrafted holistic image descriptors. We demonstrate the suitability of our method for real-time scene categorisation in autonomous driving.

The work described in this chapter was published as *Fast and Efficient Scene Categorisation for Autonomous Driving using VAEs* [210] at the 2022 Irish Machine Vision and Image Processing (IMVIP) conference.

5.2 Related Work

The success of deep learning in the field of computer vision over the past decade has resulted in dramatic improvements in performance in areas such as object recognition, detection, and semantic segmentation. However, the performance of scene recognition is still not sufficient to some extent because of complex configurations [211].

Early work on scene categorisation includes [19] where the authors proposed a computational model for the recognition of real-world scenes that bypasses the segmentation and the processing of individual objects or regions. They use a set of perceptual dimensions — namely, naturalness, openness, roughness, expansion, and ruggedness — that represent the dominant spatial structure of a scene, estimated using spectral and coarsely localised information. However, such methods were soon dominated by the introduction of global descriptor based methods due to their lack of sufficient discriminability when distinguishing between complex scenes.

[1] on consumer-grade desktop with Intel i9-9900K and Nvidia RTX 2080Ti

Histogram of Oriented Gradients (HOG) [18] descriptor, originally devised for object detection, captures edge and gradient structures within an image to distinguish the shape and appearance of objects in various contexts. More recently, researchers have used Histogram of Oriented Gradients (HOG) [18] and its extensions, such as Pyramid HOG (PHOG) [50] for mapping and localisation [117]. Although these approaches have shown strong performance in constrained settings, they lack the repeatability and robustness required to deal with the challenging variability that occurs in natural scenes caused by different times of the day, weather, lighting, and seasons [186].

To overcome these issues, recent research has focussed on the use of learned global descriptors. Probably the most notable here is NetVLAD, which reformulated VLAD through the use of a deep learning architecture [67], resulting in a CNN-based feature extractor using weak supervision to learn a distance metric based on the triplet loss.

Variational Autoencoders (VAE), introduced by [88], were originally designed as generative models capable of learning latent representations and producing novel data samples, such as human face images, by approximating complex data distributions. Since the introduction of the CelebA dataset [212], multiple implementations of VAEs have shown success in generating human faces [213, 214, 215]. However, VAEs became less popular for reconstructions, as they often produce blurry, less saturated images and have been shown to lack the ability to generate high-resolution images for domains that exhibit multiple complex variations, e.g., realistic natural landscape images.

Besides their use as generative models, VAEs have also been used to derive scalar variables from images in the context of autonomous driving, such as for vehicle control [215]. More recently, [216] used a VAE to generate a soiling mask region prior, which was then utilised by a GAN to simulate camera soiling in driving images. In this work, we re-examine the prospect of utilising VAEs, not for generating images but to train and capture high-level features of the scene. By focussing solely on the encoder, we extract global feature descriptors from the latent space, leveraging the VAE's ability to learn bounded, compact, and informative representations that are beneficial for precursor tasks, particularly scene categorisation.

To accelerate progress in general scene recognition, a number of researchers have developed datasets for training and/or evaluation. Examples include MIT Indoor67 [217], SUN [122], and Places 365 [124]. While these datasets capture a wide variety of

scenes, they are not well-suited for developing scene categorisation techniques specific to autonomous driving. Even when relevant categories are present, the images often do not reflect scenes typically encountered in driving scenarios. For instance, the Places365 dataset lacks a dedicated *city* category, and while the SUN dataset includes a *city* category, the images within it do not represent realistic driving environments, limiting their applicability to autonomous driving tasks. Given this, in our research we choose to utilise images from large public sequential driving datasets such as Oxford RobotCar [7] in an unsupervised manner and curate our own evaluation datasets targeted at our domain of interest.

In this chapter, we propose training a VAE with a reconstruction task, utilising its encoder to map images to a multi-dimensional latent space and using the latent vectors as compact embeddings that serve directly as global descriptors for images. To the best of our knowledge, this is the first time VAE latent vectors are used as global image descriptors. In detail, we train a convolutional VAE in an unsupervised manner with images from the Oxford RobotCar dataset [7] that exhibit strong visual changes caused by seasons, weather, time of the day, etc., and use the latent vectors inferred using the encoder as global descriptors. We show using experiments that the VAE encoder captures high-level features of the image, producing a mapping in a multi-dimensional standard normal latent space. We then use a simple linear classification head trained with a small manually labelled dataset with fewer than 500 images to map the global descriptors to the required scene categories: rural, urban, and suburban.

5.3 Methodology

We begin by presenting an overview of the proposed architecture in Figure 5.2, followed by a detailed explanation of the individual methodologies employed in our approach.

5.3.1 Why VAE?

To capture high-level scene features, we propose using a reconstruction task as the primary loss signal, coupled with a bottleneck architecture to produce compact intermediate representations. To meet these requirements, we employ a Variational Autoencoder

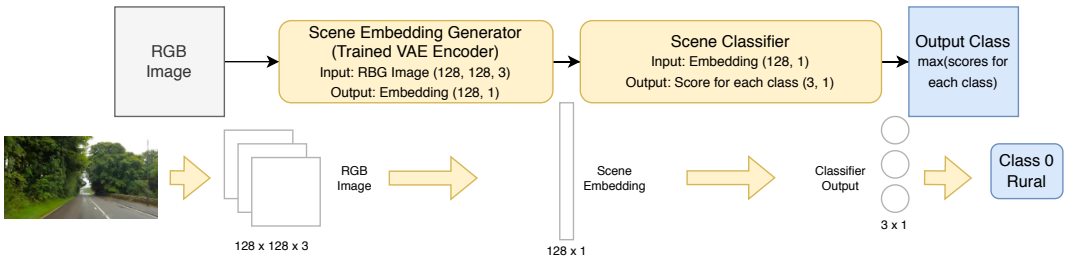


Figure 5.2: Overall system architecture for scene categorisation. Input and output for each module are shown in the bottom row, including an example.

(VAE), which provides the desired functionality while maintaining efficiency during inference. The rationale behind these design choices is explained in detail below.

Reconstruction task: We hypothesise that the reconstruction task can be efficiently used to capture high-level information in images. In order to support this hypothesis, we employ the deep image prior [218] approach, where an untrained CNN is tasked with reconstructing an image. To this end, we utilise a CNN architecture consisting of three convolutional blocks, followed by a projection layer that reduces the dimensionality to a 128-dimensional embedding. This is followed by three upsampling convolutional blocks. Each convolutional block contains two stacked ReLU-activated convolutional layers. The network undergoes multiple iterations of backpropagating the reconstruction loss as it is trained on the same image until a desired level of reconstruction is achieved.

Notably, when noisy input images (e.g., with uniform or salt-and-pepper noise) are used, the network does not prioritise modelling the noise, effectively allowing it to function as an image denoiser. Additionally, finer details in the image, such as pedestrians or cars, are given less importance during reconstruction, as reproducing these elements results in only a marginal reduction in the overall loss. The reconstruction loss is significantly reduced when the network captures and reproduces the high-level visual elements of the image, such as the overall scene layout and prominent regions including the sky, road, buildings, and large objects. Consequently, the network's emphasis on high-level features leads to a more efficient and accurate reconstruction of the broader structure of the scene while disregarding finer, less critical details. This is further evidenced by its use in image inpainting, as the network naturally fills in small missing patches, even without being explicitly tasked to do so, in its effort to reconstruct the image as a whole. This behaviour is evidenced by the reconstructed images presented in Figure 5.3, highlighting the network's ability to prioritise and

retain high-level scene information while disregarding less relevant details using the reconstruction task.

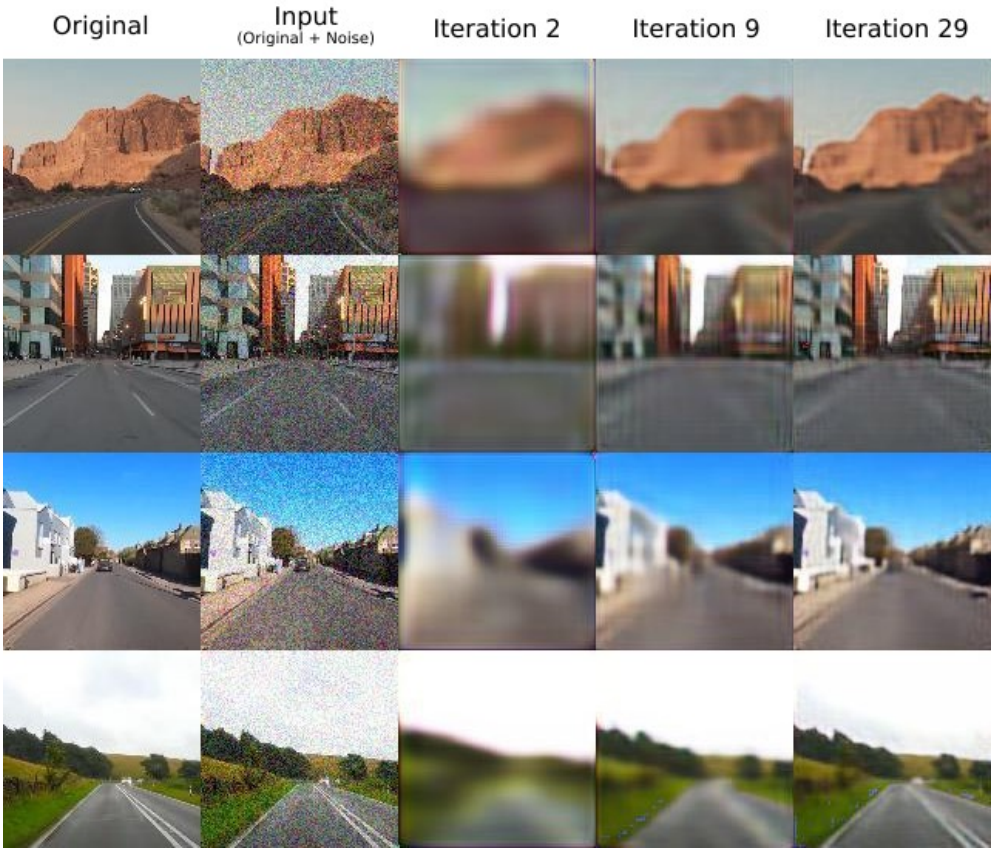
However, this method cannot be directly employed for scene categorisation due to the requirement of training a new network from scratch for each image. Although it does not require pretraining, the need for multiple iterations of backpropagation for each image introduces significant computational overhead during its use, making it impractical for realtime applications. While it is possible to adopt a pretrained CNN architecture involving a bottleneck similar to the one used above, tasked with reconstruction, we specifically opt to use a VAE for the reasons outlined below.

Interpretable Intermediate Representations: VAEs encode data as multivariate distributions rather than as a single point in the latent space, facilitating sampling and interpolation. The randomness introduced in the encoding process makes the latent space continuous, enabling the extraction of interpretable intermediate representations of the data. This allows the inferred latent variables to be used as global descriptors, capturing the high-level visual information necessary for scene categorisation. Moreover, these global descriptors can be utilised in VPR systems beyond scene categorisation, serving as valuable inputs for other precursor tasks. Specifically, we employ them for visual localisation, a topic that will be explored in detail in the next chapter.

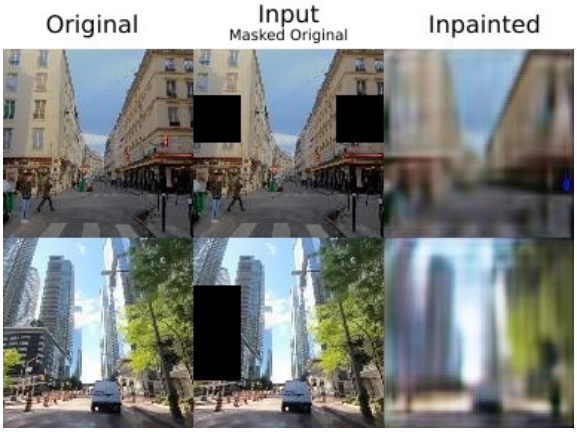
Constrained Latent Space: Mathematically, in VAEs, the objective is to approximate the true posterior distribution of the latent variables z, given the input data x, through the learned encoder distribution $q_\phi(z|x)$, where ϕ represents the parameters of the encoder. Since estimating the true posterior $p_\theta(z|x)$ (θ being parameters of the decoder) is intractable due to the complexity of integrating over all possible latent variables, variational inference is employed to approximate it. To achieve this, the VAE employs a loss function that consists of two key components, the reconstruction loss and the regularisation term associated with Kullback-Leibler (KL) divergence, given by the following equation:

$$\mathrm{L}(x;\phi,\theta) = -\mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] + \mathrm{D}_{\mathrm{KL}}(q_\phi(z|x)||p_\theta(z)) \tag{5.1}$$

where, $\mathbb{E}$ is the expectation operator to calculate the expected log likelihood of the data x given latent variables z, where z is sampled from the distribution $q_\phi(z|x)$, and D_{KL} quantifies the difference between the variational distribution (or the encoder



(a) Reconstructions of denoised images at different iterations.



(b) Inpainting of masked patches

Figure 5.3: Example driving images reconstructed using the deep image prior approach [218] from an untrained CNN architecture with a bottleneck, demonstrating high-level features such as sky, buildings, roads, and vegetation being retained, while finer details of smaller objects such as pedestrians and cars are less apparent. *Original* images are from the Scene Categorisation dataset, explained later in Section 5.3.4.

distribution) $q_\phi(z|x)$ and the prior distribution $p_\theta(z)$.

Thus, the reconstruction loss (first term) measures how well the decoder can recreate the input data x from the latent representation z, while the KL divergence (second term) constrains the learned latent distribution $q_\phi(z|x)$ to be similar to the assumed standard normal Gaussian prior distribution $p(z)$. Thus, VAE learns latent representations that are not only informative for reconstructing the input data but also conform to the desired prior distribution, ensuring that each latent variable lies within a predictable range. In contrast, features extracted from common image feature extractors, such as CNNs and transformers, lack such inherent constraints on their values, making the latent space of the VAE more structured and interpretable for downstream tasks. As such, systems that generate such intermediate representations, rather than directly mapping pixels to outputs, tend to demonstrate superior task performance and improved generalisation to previously unseen environments [201].

5.3.2 VAE Design

To develop a scene categorisation pipeline utilising a VAE model capable of real-time performance, we adopt the use of LeakyReLU [219] activated strided convolutional layers and transposed convolutional layers coupled with Batch Normalisation [220] in the encoder and decoder VAE convolutional blocks, respectively. The convolution block, comprising strided convolution, BatchNorm, and LeakyReLU, is fused into a single computational unit during deployment by combining their mathematical operations and parameters. This fusion reduces memory usage by eliminating intermediate tensors between operations, lowers computational overhead, minimises memory access operations, and improves inference speed by reducing layer transitions. Moreover, this design eliminates the need for max pooling and other computationally expensive aggregation layers, enabling deployment advantages on convolutional accelerators, ASICs (Application Specific Integrated Circuits), and automotive-grade SoCs (System on Chip) widely used in production vehicles that support ADAS (Advanced Driver Assistance Systems) features. Specifically, it allows for accelerated inference with reduced latency, allowing for efficient real-time operation.

We utilise an encoder consisting of six convolutional layers with output channels of 32, 64, 128, 256, 512, and 1024, followed by a projection layer, which reduces the

representation to a 128-dimensional latent vector. We employ a symmetric decoder, where the Conv2D operations are replaced with ConvTranspose2D operations to upsample the feature maps and reconstruct the input image. We then train this VAE [88] model on the reconstruction task using three traversals of the main route in the Oxford RobotCar dataset, which exhibit variations due to changes in season and time of day: *2014-12-09-13-21-02* (Winter Day), *2014-12-10-18-10-50* (Winter Night), and *2015-05-19-14-06-38* (Summer Day). Given that there are approximately 30000 images in each traversal, we subsample each traversal using the sequential adaptive sampling strategy (see Section 3.4.3.1) with parameters $\tau_{d_{\text{acc}}} = 5\,\text{m}$ and $\tau_{\theta_{\text{acc}}} = 15°$ to obtain 1787, 1879 and 1825 visually dissimilar images, respectively.

We train the VAE from scratch, opting not to use pretrained weights from ImageNet [96] on widely used architectures such as ResNet [58]. This decision is based on the fact that ImageNet is primarily designed to learn features specific to object-centred images. In contrast, our goal is to capture high-level visual scene information rather than fine-grained details of individual objects. To this end, we resize the images to 64×64, deliberately avoiding the learning of fine features related to discrete objects such as cars or pedestrians. This reduced resolution also enables faster training due to the smaller image size. We use a constant learning rate of 0.005 and do not apply weight decay during training. The model is trained for up to 500 epochs, with early stopping triggered if validation loss does not improve for 100 consecutive epochs. This configuration allows sufficient training time for convergence while avoiding unnecessary computation once improvements plateau.

After training is complete, we conduct a qualitative evaluation by manually inspecting the reconstructed images. In more detail, we focus on assessing whether the reconstructions effectively capture high-level scene details, ensuring that primary elements such as roads, buildings, sky, and vegetation are well-represented, while finer details of individual objects are intentionally minimised. While we anticipate some level of blur due to the inherent limitations of VAEs, the reconstructions should remain sufficiently clear to distinguish key elements of the scene, like roads, buildings, and the sky. The evaluation also includes examining reconstructions across different times of day and seasons. Additionally, we examine and manipulate the latent variables to interpret and evaluate the intermediate representations. This process helps us assess the model's ability to capture meaningful variations in the scenes, allowing for a deeper understanding of how different aspects of the scene are encoded and reconstructed.

5.3.3 Scene Embedding

The original VAE [88], i.e., the Vanilla VAE, has since seen numerous improvements and the development of various extensions. Each variant introduces different modifications, often by adding terms to the loss function to enhance specific aspects of the model's performance. For our experiments, we select several of the most widely recognised VAE variants,

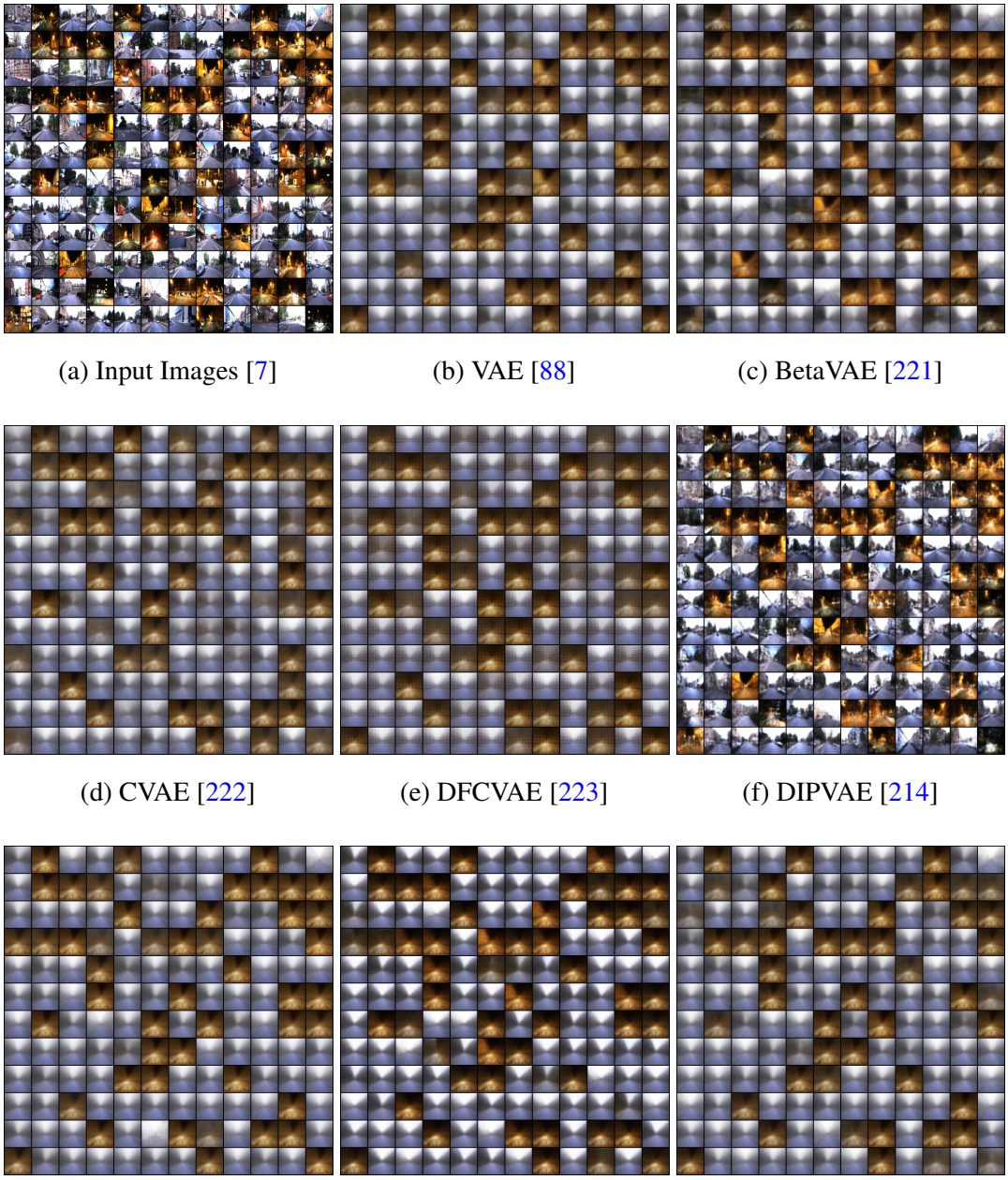
- BetaVAE [221]
- CategoricalVAE [222]
- DFCVAE [223]
- DIPVAE [214]
- InfoVAE [213]
- LogCoshVAE [224]
- MIWAE [225]

and train them under identical conditions, using the same dataset, training procedures, and evaluation metrics as previously described.

Among these variants, DIPVAE (Disentangled Inferred Prior VAE) [214] stands out, producing less blurry reconstructions with higher visual fidelity and greater diversity across different input-output pairs; see Figure 5.4. DIPVAE achieves this by introducing a disentanglement regulariser over the inferred prior, promoting better separation of latent variables without compromising the overall quality of the reconstructions. While β-VAE [221] also focuses on disentangling latent representations, in DIPVAE there is no extra conflict introduced between disentanglement of the latents and the observed data likelihood, resulting in better generalisation. As such, the disentangling of features in the latent space minimises overlap across dimensions, resulting in more interpretable and meaningful intermediate representations.

We also trained the DIPVAE model on 128×128 images from the same dataset, with the embedding dimension kept fixed at 128. This increase in resolution led to slightly more detailed reconstructions without altering the fundamental structure of the latent representation, and this 128×128 variant is used throughout this chapter.

We utilise the trained DIPVAE encoder to infer disentangled latent variables from input images, which serve as compact global descriptor embeddings. While the VAE decoder contributes to the loss function during training, it is discarded during inference,



(a) Input Images [7] (b) VAE [88] (c) BetaVAE [221]

(d) CVAE [222] (e) DFCVAE [223] (f) DIPVAE [214]

(g) InfoVAE [213] (h) LogCoshVAE [224] (i) MIWAEVAE [225]

Figure 5.4: Reconstructions of different variants of VAE trained on images from Oxford RobotCar [7]

Rural		Suburban		Urban	
Jarrahdale Perth	33	Hawaii	8	Indianapolis	30
Missouri Ozarks	22	Howth	17	Nashville	21
Southern Illinois	43	Melbourne	33	Paris	24
Stockport Buxton	25	Stockport Buxton	17	St Louis	52
Utah	75	Wimbledon	21	Toronto	33
Rural	198	Suburban	96	Urban	160
Total					**454**

Table 5.1: Information about our Scene Categorisation dataset

as it is not required for obtaining these global descriptors. Given the focus on capturing high-level scene information and the evaluation strategy employed, we employ these global descriptors as scene embeddings.

We further note that we extensively use these global descriptors in Chapter 6, where they play a key role in establishing the hierarchical representation used in visual localisation and mapping.

5.3.4 Scene Classifier

Although trained to capture high-level scene features through an unsupervised reconstruction task, the scene embeddings lack a direct correspondence to specific semantic categories: urban, rural, or suburban. To bridge this gap, we introduce a lightweight supervised classification head, implemented as a simple linear dense layer without an activation function, to map these embeddings to the desired scene categories.

The training of this classification head requires a small, purpose-built dataset tailored to driving scenarios. To this end, we curate a custom Scene Categorisation Image Dataset[2] (SCID), for training and testing the classifier by manually selecting screenshots from driving videos available on YouTube, taken at various timestamps. The screenshots are selected to be well-spaced in time within the videos, ensuring to avoid choosing images from the same region. Each image is assigned to one of three scene categories:

[2]Dataset available to download from https://gist.github.com/saravanabalagi/1cda6ae06c4cf722fd2227e83eadc792

Figure 5.5: Random samples from the SCID dataset. Top Row: Rural, Middle Row: Suburban, Bottom Row: Urban. Best viewed zoomed on a computer screen.

rural, urban, or suburban. Images labelled as rural contain no buildings and represent open, unambiguous countryside settings. Urban scenes are characterised by densely packed buildings, including multistorey structures and skyscrapers, while suburban scenes depict housing estates near cities with more sparsely distributed houses. Only images that clearly and definitively fit one of these categories are selected; ambiguous cases such as rural areas with a few houses that resemble suburban environments are excluded. Each category includes images captured from various cities and regions to ensure diversity in the dataset; see Table 5.1. Some examples are shown in Figure 5.5, and as can be seen, the dataset covers a variety of landscapes, including desert and mountainous landscapes, and exhibits mild to moderate illumination and seasonal changes such as fallen leaves and different times of the day.

The SCID training split comprises two-thirds of the images randomly selected from each route, yielding a total of 314 images. Given that the images are non-sequential and visually dissimilar, representing distinct locations, this random sampling does not result in an information leak between the train and test sets. A linear classifier is then trained on this set. Training is done until there is no further decrease in loss for 10 consecutive epochs, indicating convergence.

5.4 Experiments and Evaluation

To verify the suitability of the embeddings for scene categorisation, we use the widely used evaluation procedure employed to test embeddings for classification tasks [96, 101].

Since our proposed architecture employs compact intermediate representations as input to the linear classifier for scene categorisation, we evaluate the scene embeddings directly. No additional pooling or aggregation layers are introduced, allowing us to assess the performance of the features as generated by the encoder without further modifications.

The output of the linear classifier is tested on the SCID test split containing the remaining 140 images, and the test accuracy is used as a proxy for the representation quality of the embeddings used. Evaluation was done on an Intel i9-9900K (8 cores @3.60 GHz) and Nvidia RTX 2080 Ti, and all images were resized to 128×128. We compare the results with the following benchmark learned and handcrafted holistic image descriptors: (1) NetVLAD[3] [67], a weakly supervised CNN with generalised VLAD (Vector of Locally Aggregated Descriptors) layer. (2) PHOG[4] [50], Pyramid Histogram of Gradients. For the evaluation, we consider the following candidates:

- NetVLAD 4096 dimensions: Supervised, pretrained on Pittsburgh dataset[5]
- NetVLAD 128 dimensions: Supervised, pretrained, same as above, PCA + cropped to 128 dimensions + L2-normalised, derived from NetVLAD 4096 embedding
- PHOG 1260 dimensions: Handcrafted, 60 bins and 3 levels [117]
- DIPVAE[6] 128 dimensions: Unsupervised, pretrained on 128×128 Oxford RobotCar dataset images

The experimental results are shown in Table 5.2. As expected, the supervised techniques score higher than the unsupervised and handcrafted techniques. NetVLAD 4096 shows the best performance in the evaluation with 99.29% accuracy, followed by NetVLAD 128 with 94.29% accuracy. The high accuracy is the result of (1) the technique's use of supervised learning, (2) the embedding length of 4096 allows capturing more information about the scene, and (3) NetVLAD Cropped (128 dimensions) is derived from NetVLAD 4096 through PCA, cropping and L_2 normalising. DIPVAE (128 dimensions) achieves 82% accuracy with an embedding size that is only 10%

[3] MATLAB implementation provided by authors at https://github.com/Relja/netvlad is used

[4] Code extracted from C++ implementation provided by authors at https://github.com/emiliofidalgo/htmap is used

[5] Off-the-shelf VGG16+NetVLAD+whitening model provided at https://www.di.ens.fr/willow/research/netvlad/

[6] Our own implementation in Python 3.8 and PyTorch 1.11 (CUDA 11.3) is used

Descriptor	**Type**	**Dimensions ↓**	**Accuracy (%) ↑**	**Compute Time (μs) ↓**
Random	Trivial	4096	34.29	71.3 ± 0.0
Random	Trivial	128	28.57	2.7 ± 0.0
NetVLAD	Supervised	4096	**99.29**	27560.0 ± 230.2
NetVLAD Cropped	Supervised	**128**	94.29	27563.9 ± 230.4
PHOG	Hand-crafted	1260	84.29	123.6 ± 3.9
DIPVAE (Ours)	Unsupervised	**128**	82.86	**60.4 ± 3.0**

Table 5.2: Classification accuracy on the test split of the SCID dataset for different descriptors mentioned in Section 5.3.4. A random descriptor, constructed trivially by sampling numbers from a normal distribution, is shown at the top to provide a baseline for trivial descriptors that do not capture any relevant information from the images. Compute time is the time taken to obtain the descriptor from a decoded image loaded in memory.

Route	**Dublin**	**Vancouver**	**Wicklow**	**Redwood**
Scene Type	Urban		Rural	
Total Images	14677	64672	60509	45253
Test Images	11022	51018	47688	35483
NetVLAD 4096	93.37	96.99	99.99	99.86
NetVLAD 128	78.47	98.84	99.98	99.87
PHOG 1260	91.25	98.50	88.46	86.60
DIPVAE 128	95.70	83.72	99.44	95.60

Table 5.3: Classification accuracy (%) on the larger SCVD dataset using various descriptors. Total and test image counts are provided in the top two rows.

the size of PHOG (1260 dimensions) and 3.1% the size of the NetVLAD 4096. We note that both NetVLAD and DIPVAE use GPU acceleration, while PHOG uses CPU optimisations and multi-threading. Notably, DIPVAE is computed more than twice as fast as PHOG and several orders of magnitude faster than NetVLAD. Performance variance is less significant given the relatively small size of the dataset; however, the substantial reduction in runtime for models achieving comparable accuracy represents a considerable practical advantage.

We further evaluate the linear classifiers on a second video-based Scene Categorisation Video Dataset[2] (SCVD). Here, we utilised frames from a variety of extended driving videos collected from YouTube, as shown in Table 5.3. Each video was labelled as a single scene category, where collectively this resulted in a total of over 185k images. On each route, we first remove the first 900 frames (30 seconds at 30 fps) to avoid encountering intro text, crossfades, and other effects. Then, we use the first 20% of the frames (~40k) for training and the rest 80% (~145k) for evaluation.

Due to the nature of the source, some portions of these sequences may not align strictly with the intended scene categories, potentially introducing noise into the results. For instance, certain frames from videos labelled as urban may visually resemble suburban and, in some cases, even rural environments. However, based on the length of each sequence and manual inspection, we estimate that more than three-quarters of the images in the Rural (Wicklow and Redwood) and City (Dublin and Vancouver) videos are unambiguously classified as *Rural* and *Urban*, respectively. Despite this, a robust descriptor is still required to predict several thousand frames correctly and consistently, as the visual content in these videos changes dynamically. We note that the results may exhibit high variance due to the variability in scene types within the video.

Table 5.3 presents the classification accuracy of the descriptor candidates discussed earlier, evaluated on the SCVD dataset. Table 5.4 illustrates qualitative results on randomly selected images from the SCVD dataset, displaying both the ground truth (GT) labels and the predictions of the models being evaluated. We further perform a qualitative analysis of the results to validate the categorisation accuracy. A video showing the results of categorisation outcomes is made available online[7]. As such, DIPVAE performs consistently well and shows similar performance to that of NetVLAD and PHOG while having a much smaller embedding dimensionality and a significantly

[7]available at https://youtu.be/6a71B7yUhe8

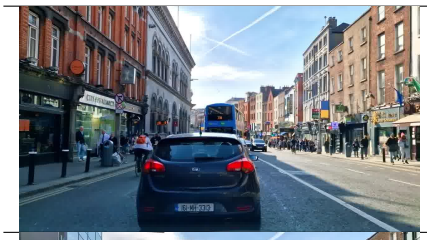
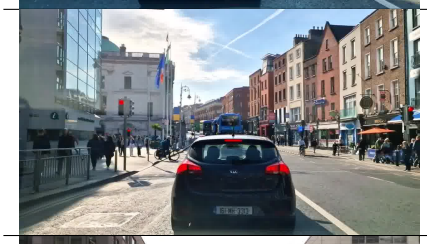
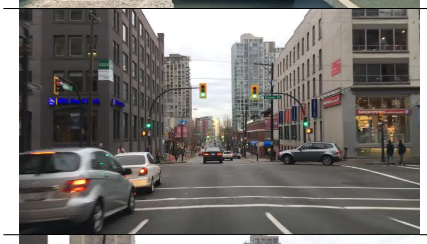
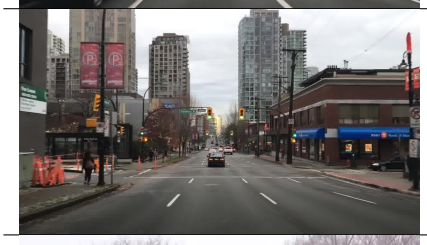
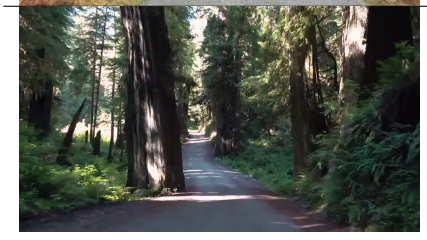
Image	GT	*NetVLAD 128*	*NetVLAD 4096*	*PHOG 1260*	*DIPVAE 128*
	Urban	Suburb	Urban	Urban	Urban
	Urban	Suburb	Urban	Urban	Urban
	Urban	Urban	Urban	Urban	Urban
	Urban	Suburb	Urban	Urban	Urban
	Rural	Suburb	Urban	Suburb	Rural
	Rural	Rural	Suburb	Rural	Rural
	Rural	Suburb	Urban	Urban	Suburb

Table 5.4: Samples from the SCVD dataset along with predictions from different models used in evaluation.

faster compute time.

5.5 Conclusion

Our proposed solution to scene categorisation uses a novel architecture made up of an unsupervised convolutional VAE encoder and a simple supervised linear classifier head trained with fewer than 500 images. We demonstrate and report results for our approach in which we use DIPVAE disentangled latent vectors directly as global descriptors coupled with a linear classifier for scene categorisation. Unlike end-to-end pixel-to-class models, our approach generates meaningful and intermediate representations interpretable through decoder visualisation of latent vectors, coefficients of which are confined to standard normal distribution during training. The experimental results demonstrate that the DIPVAE latent vectors capture high-level scene information from the image, supporting their usage as global descriptors. These global descriptors exist in a constrained continuous multi-dimensional standard normal manifold, allowing easier comparison and interpretation compared to unbounded embedding hyperspaces. The proposed global descriptor is very efficient, featuring a compact embedding length of 128, and is significantly faster to compute. It remains robust across diverse real-world conditions, capturing sufficient scene information for reliable scene categorisation, despite variations in weather (sunny, cloudy, and rainy), geographic diversity (mountains, deserts, and forested landscapes), and different times of day (daylight hours, sunrise, and sunset). Furthermore, the VAE architecture made up of standard convolutional blocks without the use of pooling and aggregation layers allows more efficient, fast, low-latency, and near real-time inference using hardware convolutional accelerators, substantiating their use in autonomous vehicles to quickly determine location context as a precursor task.

Further available information, such as GPS, together with recent predictions, could also be used to make more temporally consistent decisions about the scene category (e.g., avoiding categorising the environment as rural when driving along a tree-lined route in a city). Finally, we indicate that the proposed global descriptors, being intermediate representations, are useful for other tasks and actions that only require high-level features, including scene information present in the image. We note that there is a potential to further improve this performance using supervised and weakly supervised

techniques. If and when labels are available, the VAE backbone can be set to learn with a small learning rate (e.g., one-tenth relative to that of the head) in an end-to-end manner. In our future work, we intend to explore the potential of adding further categories, such as motorways, tunnels, and car parks, that are useful and provide more context for various autonomous driving tasks. Given the successful results, we further intend to integrate this approach in a hierarchical place recognition pipeline, where these compact global representations are used to aggregate images to facilitate faster image retrieval.

Chapter 6

Scalable and Efficient Hierarchical Mapping and Localisation

6.1 Introduction

Robot environmental maps captured using cameras are generally modelled using two representations: metric and topological. Metric maps are sensitive to noise, as they retain a large amount of information about the environment, such as distances, measures, or sizes. Metric maps are more difficult to build and maintain and are computationally demanding. Consequently, localisation approaches that maintain a full metric map on a mobile device or robot are often restricted to small-scale environments due to the high memory requirements. On the other hand, topological maps model the environment using higher-order objects and their relationships using graphs, in which the nodes represent objects or places and edges correspond to the paths. These maps are simple and compact, scale better, and require much less space to be stored than metric maps.

Visual loop closure detection is a core component of many vision-based mapping approaches, where the previously visited places are recognised using Visual Place Recognition (VPR) techniques. Typically, the image is encoded as a vector of specified length by means of a global descriptor, where a similarity metric appropriate to that type of descriptor is utilised for image comparison. Finding the closest match for a query image involves an exhaustive search through all encoded images using this metric, and consequently, the time taken to search increases linearly as the total number of images

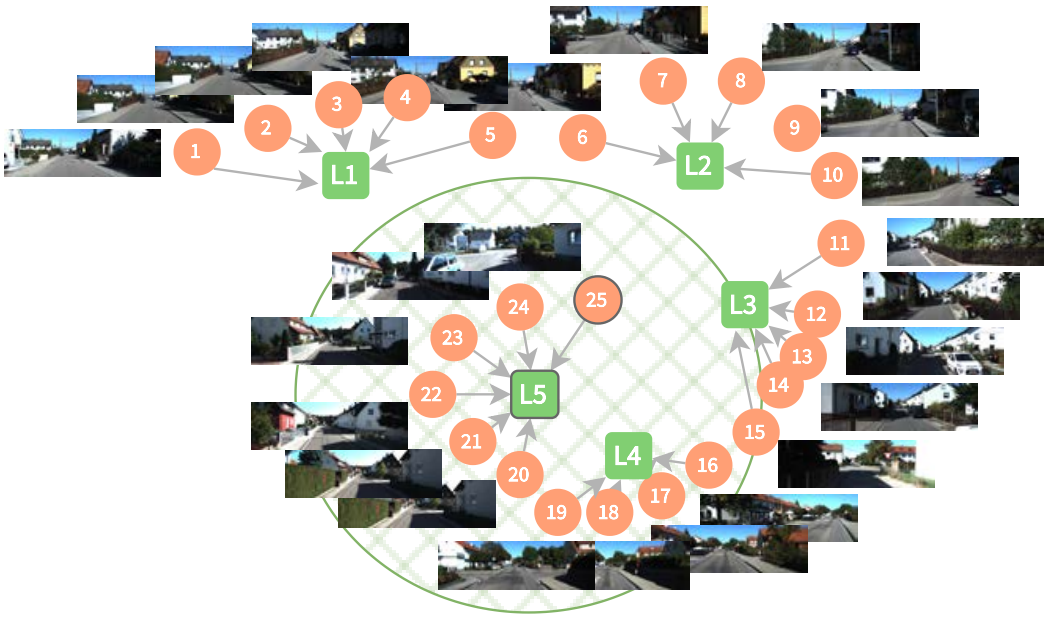


Figure 6.1: An illustration of hierarchical mapping and localisation in the global descriptor embedding space represented in 2D for clarity. Locations are represented as green bubbles, each encompassing a cluster of visually similar images, shown in orange bubbles and connected to their respective locations. As I_{25} gets processed, the hatched green circle around L_5 shows the reduced search space containing the 3 locations, only within which the loop closing image candidate is searched.

in the map increases.

An important approach to reducing this time complexity is to use indexing techniques. In the seminal work of [13], the authors proposed an approach to search viewpoint invariant region descriptors using inverted file systems and document rankings similar to the ones used in text retrieval systems. Following this work, [23] built upon popular techniques of indexing descriptors extracted from local regions and used a vocabulary tree trained in an unsupervised fashion that hierarchically quantized descriptors from image keypoints. Although such techniques offer significant advantages over exhaustive searching, they can be infeasible for mapping very large environments due to the high memory requirements and computational overhead needed to maintain and search within large vocabulary trees.

A hierarchical representation of the environment [226, 117], where images that present a similar appearance are grouped together in nodes, can significantly reduce the

search space when finding similar places. As such, the hierarchy helps accelerate the retrieval process by skipping multiple nodes that are not relevant altogether. Although such research has shown the potential benefits of hierarchical matching, limited consideration has been given to the suitability and comparative performance of different feature representations used within these approaches.

We propose to use compact learned global descriptors in hierarchical topological mapping of environments to aggregate sequences of images with similar appearance into location nodes, based on the approach first proposed by [117]. Many learned descriptors with improved retrieval accuracy have been incorporated into place recognition methods to enhance overall recall. In this chapter, we focus on addressing the challenges of scalability and efficiency, in particular, when such methods are used on longer trajectories. We show through our evaluation that the use of learned global descriptors is deemed necessary even when handcrafted global descriptors perform similarly to learned descriptors in terms of recall at 100% precision on benchmark datasets. This is due to learned descriptors' ability to improve efficiency, reduce total runtime, and minimise the total number of relevant locations searched, among other factors.

Our contributions can be summarised as follows:

- We extend the Hierarchical Topological Mapping system from [117], perform an in-depth analysis of its components, make a number of improvements to the underlying implementation, and most significantly, incorporate learned global descriptors,
- We compare hierarchical topological mapping technique with state-of-the-art hand-crafted and learned global descriptors and present results of a comprehensive evaluation of the impact of the global descriptor used,
- Through empirical analysis, we identify and define the characteristics of an ideal global descriptor supporting hierarchical matching amenable to scalable and efficient visual localisation and present a methodology for quantifying and contrasting these characteristics, and,
- We propose the use of compact learned global descriptors that excel in continuity and distinctiveness characteristics as an efficient and scalable means for hierarchical topological mapping.

This chapter has been published as *Scalable and Efficient Hierarchical Visual Topological Mapping* [227] at the 2023 IEEE International Conference on Advanced Robotics (ICAR).

6.2 Background

Hierarchical representations can significantly reduce search times within mapping and localisation. Consequently, there have been a number of recent advancements in hierarchical mapping and localisation techniques [228, 229, 230]. A notable state-of-the-art approach is HFNet [109], which follows a hierarchical approach based on a monolithic CNN that simultaneously predicts local features and global descriptors for accurate 6-DoF localisation. Although these techniques use hierarchical approaches for more efficient processing, their use of a metric representation makes them intractable to run on longer sequences that are several kilometres long.

There have also been a number of works suitable for very large-scale mapping and localisation [110, 39, 45] without using an explicit metric representation. More recently, Garcia-Fidalgo et al. [117] proposed an appearance-based approach for topological mapping based on a hierarchical decomposition of the environment where the map aggregates images with similar visual properties together into location nodes, which are represented by means of an average global descriptor and an index of local binary features.

A central decision in the development of each of the above systems is the choice of feature descriptor, given its impact on the system's performance. The traditional and handcrafted global descriptor approaches, such as [13, 39, 40] using Bag of Visual Words (BoVW), were commonly used in early visual SLAM systems. More recently, the research community has focussed on the use of learned global descriptors given their compelling performance in the field of computer vision in areas such as object recognition, detection, segmentation, and image representation [58, 231, 202, 186, 232, 233, 158].

In [67] the authors introduced NetVLAD, a generalised differentiable VLAD layer in a CNN trained end-to-end using weak supervision to learn a distance metric based on the triplet loss. Various extensions to NetVLAD have also been proposed, such

as [68, 69, 70, 71], to produce patch-level features and/or capture multi-scale features.

There have also been many attempts to improve the performance of image retrieval systems using semantic information. [234] presents a method for scoring the individual correspondences by exploiting semantic information about the query image and the scene to perform a semantic consistency check useful for outlier rejection. [85] proposed the Local Semantic Tensor (LoST) descriptor derived from the convolutional feature maps of a pretrained dense semantic segmentation network.

More recently, [210] use a generalised global descriptor that captures high-level scene information from the image using an unsupervised convolutional Disentangled Inferred Prior Variational Autoencoder (DIPVAE) [214], to map images to a multi-dimensional latent space.

As such, several learned image descriptors have emerged in the research field, with a primary focus on capturing finer image features and enhancing retrieval accuracy. However, their ability to scale to visual place recognition over longer trajectories and, in particular, their efficiency in a hierarchical setup has not been adequately evaluated. In this chapter we address this issue, proposing the adoption of a global image descriptor that enables image aggregation into locations and minimises the number of searches of prior locations for achieving scalable and efficient place recognition. Our approach is to extend the hierarchical topological mapping algorithm proposed by [117] incorporating the use of learned global descriptors for representing locations. We perform extensive analysis on the impact of the global descriptor on the formation of locations, their discriminability, and the coherence of images within locations, which in turn affects the overall recall and runtime of the algorithm. Through our analysis, we identify a set of required characteristics for feature representation to support scalability and efficiency in hierarchical mapping.

6.3 Methodology

We extend the Hierarchical Topological Mapping (HTMap) algorithm proposed by [117] to allow us to evaluate the performance of a variety of learned feature representations. For completeness, we first provide a summary of the relevant elements of the HTMap algorithm here.

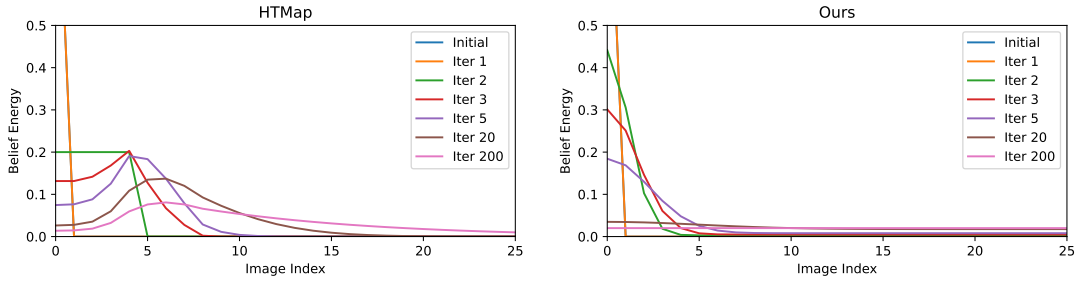


Figure 6.2: Plots showing beliefs after repeated posterior calculation (involving energy diffusion and normalisation) initialised on a set of beliefs for 100 images. Left: Original HTMap [117]: beliefs are not diffused even after 200 iterations; Right: Ours: beliefs are diffused significantly at 20 iterations and completely at 200 iterations.

The HTMap algorithm creates a topological map of locations, where each location L is an aggregation of similar images. Formally, $L = \{I \mid \forall J \in L,\, d(I_{\text{gd}}, J_{\text{gd}}) < t_{\text{nn}}\}$, where the global image descriptor I_{gd} encodes a holistic representation of the image, I and J are images, d is the distance function, and t_{nn} is the aggregation threshold. Each location also maintains its own location descriptor L_{gd} defined as a function of its associated images. This aggregation into a single location descriptor L_{gd} creates a two-level hierarchy, as shown in Figure 6.1, allowing newly processed images to be compared directly with the higher-level locations. Loop closing image candidates are then searched only within the set $\{L \mid 1 - d_n(I_{\text{gd}}, L_{\text{gd}}) > t_{\text{llc}}\}$, where t_{llc} is the location loop closure threshold, and d_n being distance min-max normalised across all locations. In [117], Pyramid Histogram of Oriented Gradients (PHOG) [50] is used as the global descriptor to compute I_{gd} with Chi-square distance as the distance function d. For a comprehensive explanation of the HTMap algorithm, the reader is advised to refer to [117].

We fork the original HTMap implementation[1] and add a number of alterations, enhancements, and optimisations to the pipeline for more accurate and efficient localisation. We fix a number of implementation errors with preloading descriptors from disc, NaN values when calculating distances and normalised similarities, and memory optimisation, amongst others. Then, we conduct an in-depth analysis of the performance of individual components constituting the system.

HTMap adopts a discrete Bayes filter utilising an evolution transition model to

[1] available at https://github.com/emiliofidalgo/htmap

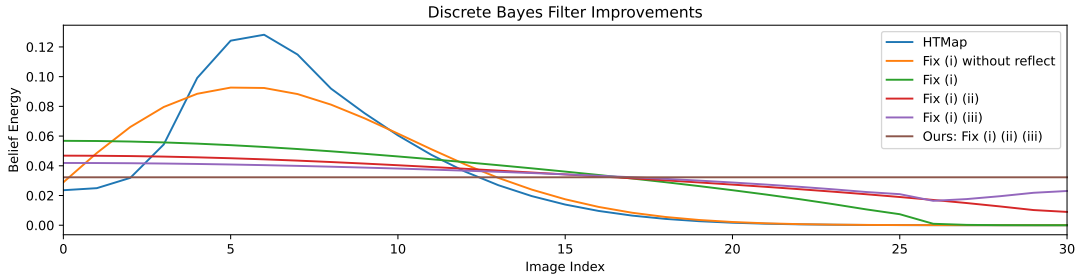


Figure 6.3: Fixes to discrete Bayes filter ablated: The plot shows the beliefs after processing 30 poses in a trial started with an initial belief of 1 for the first image and where subsequent poses do not update priors. See Section 6.3 for explanations for fixes (i) to (iii).

compute a belief distribution over the set of previously processed images. In the original implementation of HTMap, we identified problems in how the energy is accumulated and dissipated over time by the evolution model, as shown in Figure 6.2. In particular, in the absence of measurement updates (i.e., input images), the posterior distribution should eventually distribute its belief uniformly across all poses. Instead, as shown in the figure, a peak is maintained around the early poses of the sequence. We attribute these problems to three main issues: (i) The probability mass adversely accumulates energy around the initial few poses, in addition to gradually shifting away from the first pose. (ii) While calculating the posterior, after dissipating 90% of the energy to the 8 neighbours, the remaining 10% of the energy is distributed to all but 8 neighbours, resulting in irregular energy distribution. (iii) As new poses are introduced, they are initialised to have zero prior belief, leading to the persistence of accumulated energy around the initial set of poses.

To address these issues, we propose to apply 90% diffusion using a discrete Gaussian kernel summing to 0.9 with reflect padding to fix (i). To rectify (ii), we distribute the remainder 10% of the energy to all poses. Finally, to fix (iii), we initialise new poses with a value of $1/n$, where n is the number of poses already processed. Figure 6.3 ablates the fixes made, showing how our model correctly produces a final uniform belief in the absence of measurement updates.

Although the above changes resulted in an improvement in the recall of loop closures in the early stages of the mapping session, the overall recall of loop closures exhibited only a marginal increase. On further inspection, we discovered that the problems

caused by the issues were masked by the robustness of the local descriptor-based image matching module, which returns a significantly higher matching score for true image matches. However, this resilience comes at the expense of a considerable increase in runtime, as the module had to process additional locations erroneously updated to have higher beliefs.

More importantly, we adapt the framework to utilise learned global descriptors. For the histogram-based global descriptor PHOG, as a new image I is added to the active location, the location descriptor is updated as the mean of the active location descriptor L_{gd} and the global descriptor of the new image I_{gd}. For learned descriptors, we follow the same approach, replacing the Chi-square distance with the Euclidean distance.

For computing the global descriptor I_{gd} within HTMap, we utilise NetVLAD [67], LoST [85], and DIPVAE [210], representing state-of-the-art approaches in supervised learning, supervised semantic segmentation, and unsupervised variational autoencoded latent paradigms, respectively, for driving scene image analysis and feature extraction. Several recent variations of NetVLAD, such as Patch-NetVLAD [69] and Patch-NetVLAD+ [70], have been proposed to improve recall performance by providing patch descriptors in addition to the global descriptor. However, our approach only utilises the global descriptor for image aggregation, and therefore, we do not employ these variants. Additionally, while it could be argued that other NetVLAD variants, such as SPE-NetVLAD [68] and MultiRes-NetVLAD [71], could also be used, our primary objective is not to improve recall performance. Instead, our main focus is on assessing the efficacy of a global descriptor in facilitating image aggregation and minimising location searches for place recognition. As a result, in our work, we aim to utilise global descriptors that capture features using different learning approaches.

The integration of such neural networks into the framework presents a set of challenges stemming from variations in programming languages, GPU library constraints, and the potential for increased codebase complexity and tight coupling. ONNX [235] offers a standardised approach to integrating deep learning models. However, it can be limiting in cases where the neural networks utilise functions and custom operations not supported by the ONNX library. Hence, we implement a mechanism to obtain global descriptors through two approaches: (i) calling an external function for real-time online purposes, and (ii) loading pre-computed descriptors from disc for offline scenarios when available. This ensures that the framework can effectively accommodate diverse models

Dataset	# Imgs	Resolution (px)	Rate (Hz)	Dist (km)
City Centre	1237	1280 × 480	0.5	2.0
KITTI 00	4541	1241 × 376	10	3.7
KITTI 05	2761	1226 × 370	10	2.2
KITTI 06	1101	1226 × 370	10	1.2
St Lucia	21815	640 × 480	15	17.6

Table 6.1: Datasets used for evaluation

GDescriptor	Len ↓	Type	Device	BSize ↑	Compute Time (s) ↓
PHOG	1260	Handcrafted	CPU	16	0.005455 | 0.007601
LoST	6144	Supervised	GPU	22	0.181668 | 0.231041
NetVLAD	4096	Supervised	GPU	22	0.027397 | 0.048229
NetVLAD Cropped	**128**	Supervised	GPU	22	0.027907 | 0.049699
DIPVAE R128	**128**	Unsupervised	GPU	1500	0.000008 | 0.000179
DIPVAE R64	**128**	Unsupervised	GPU	**6000**	**0.000002 | 0.000040**

Table 6.2: Global Descriptors considered for evaluation. Compute times (per image) are reported for the specified max batch size *BSize* and also for a batch size of 1 separated by a vertical bar.

without being tightly bound to any specific implementation.

6.4 Experiments

In our experimental evaluation we employed the following benchmark datasets, similar to those used in [117]: City Centre [110], KITTI [126] (Sequences 00, 05 and 06) and St Lucia [46] (Sequence 19-08-09 08:45). For City Centre, images from left and right cameras are horizontally concatenated. For KITTI, RGB images from the middle camera (*Cam 2*) are used. More information about the datasets is given in Table 6.1.

As part of the in-depth analysis, we extensively utilise OdoViz [24] to visualise the image-image ground truth loop closures, superimposed on the trajectory pose information for each dataset. Through this process, we have identified and corrected

several erroneous entries in the ground truth loop closure (GTLC) matrices as provided by [236] and [117]. Corrected ground truth loops are made publicly available[2].

Furthermore, we analysed the margin criterion $m = 10$ frames used to determine a predicted loop closure image as a true positive (TP) in the original HTMap algorithm. In particular, we use OdoViz to perform a manual visual analysis of the datasets and identify that $m = 10$ is overly conservative for the St Lucia [46] dataset given its higher frame rate and the relatively higher speed of the ego-vehicle. To ensure 100% precision, $m = 50$ for St Lucia and $m = 10$ for all other datasets is used.

Table 6.2 provides information regarding the global descriptors used. For PHOG, we use the code extracted from the C++ implementation provided by authors of [117]. For NetVLAD, the MATLAB implementation of the off-the-shelf *VGG16 + NetVLAD + whitening* model pretrained on the Pitts30k [129] dataset provided by the authors of [67] is used. While a NetVLAD model pretrained on the Tokyo 24/7 dataset [131] was also available, we opted for the Pitts30k-pretrained NetVLAD for its superior matching accuracy. We additionally use a NetVLAD 128 variant where we crop and L_2 normalise the NetVLAD 4096-dimensional descriptor to 128 dimensions. For LoST, we use the MATLAB implementation of RefineNet [86] pretrained on CityScapes [132] and the Python implementation of the LoST descriptor from RefineNet embeddings provided by the authors of [85]. For DIPVAE, the PyTorch implementation with the architecture pretrained on the Oxford RobotCar dataset described in Section 5.3.4 is used. In addition to the DIPVAE variant pretrained on 128×128 images, we also include the 64×64 variant introduced in Section 5.3.3, to analyse the impact of reduced input resolution on recall performance. These variants will henceforth be referred to as R128 and R64, respectively.

Experiments were carried out on a PC equipped with an Intel i9-9900K (8 cores @ 3.60 GHz) with 32 GiB DDR4 RAM and a single Nvidia GeForce RTX 2080 Ti with 11 GiB DDR6 VRAM. The HTMap algorithm is run with a 32 byte LDB local descriptor and t_{inliers} set to 75, 80, 80, 125, and 75 inliers for City Centre, KITTI 00, 05, 06, and St Lucia, respectively, to obtain 100% precision, i.e., zero false loop closures. Other default settings, as suggested in [117], are used, and HTMap is run multiple times by varying the t_{nn} parameter to obtain a different number of locations on each dataset.

[2] available at https://github.com/saravanabalagi/htmap_gt_loops

The min and max t_{nn} are chosen such that they yield an upper bound of 250 locations (700 for St Lucia) and a lower bound of 10 locations (100 for St Lucia), respectively. The bounds and the scale of t_{nn} varies significantly for different global descriptors. Hence, for each global descriptor, we plot recall at 100% precision and the total time taken against the number of locations obtained in the map; see Figure 6.4. Each curve in the graph corresponds to a global descriptor, and each data point corresponds to a single execution of the algorithm with a specific t_{nn}. We also plot the number of False Positive Location Candidates (FPLC) proposed, which is an inverse measure of the number of relevant locations searched. As such, more FPLC proposals require searching within indices of multiple locations and would significantly contribute to increased runtimes.

Additionally, to measure the loss of recall due to the use of the hierarchical representation, we run the same set of experiments with ground truth loop closure location proposals. This results in zero FPLC and hence runs with the least total runtime. Usually, a drop in recall is expected as a result of not proposing correct loop-closing ground truth locations. However, in some cases, we also noticed a slight increase in recall. We find that certain loop closure misses cause the formation of new locations amenable to further loop closures (similar to fragmentation shown in Figure 6.5), resulting in a small increase in recall. Hence, we do not use the loss in recall using ground truth location proposals as a metric to measure performance.

Furthermore, the descriptor compute time for each global descriptor is recorded and compared. To ensure fair comparison, Python implementations[3] of the descriptor models were used. Table 6.2 shows batch and individual compute times measured for each global descriptor along with the inference device used. Batch compute times are measured with the maximum batch size *BSize* possible (limited by memory capacity and/or CPU cores), and the mean compute time per image is reported. We also report compute times for the single image inference (i.e., batch size of 1). The first image from the City Centre evaluation dataset was used for inference to measure the compute times of all models.

[3]Implementations available at:
PHOG: https://github.com/saravanabalagi/phog,
NetVLAD: https://github.com/Nanne/pytorch-NetVlad,
LoST: https://github.com/DrSleep/refinenet-pytorch

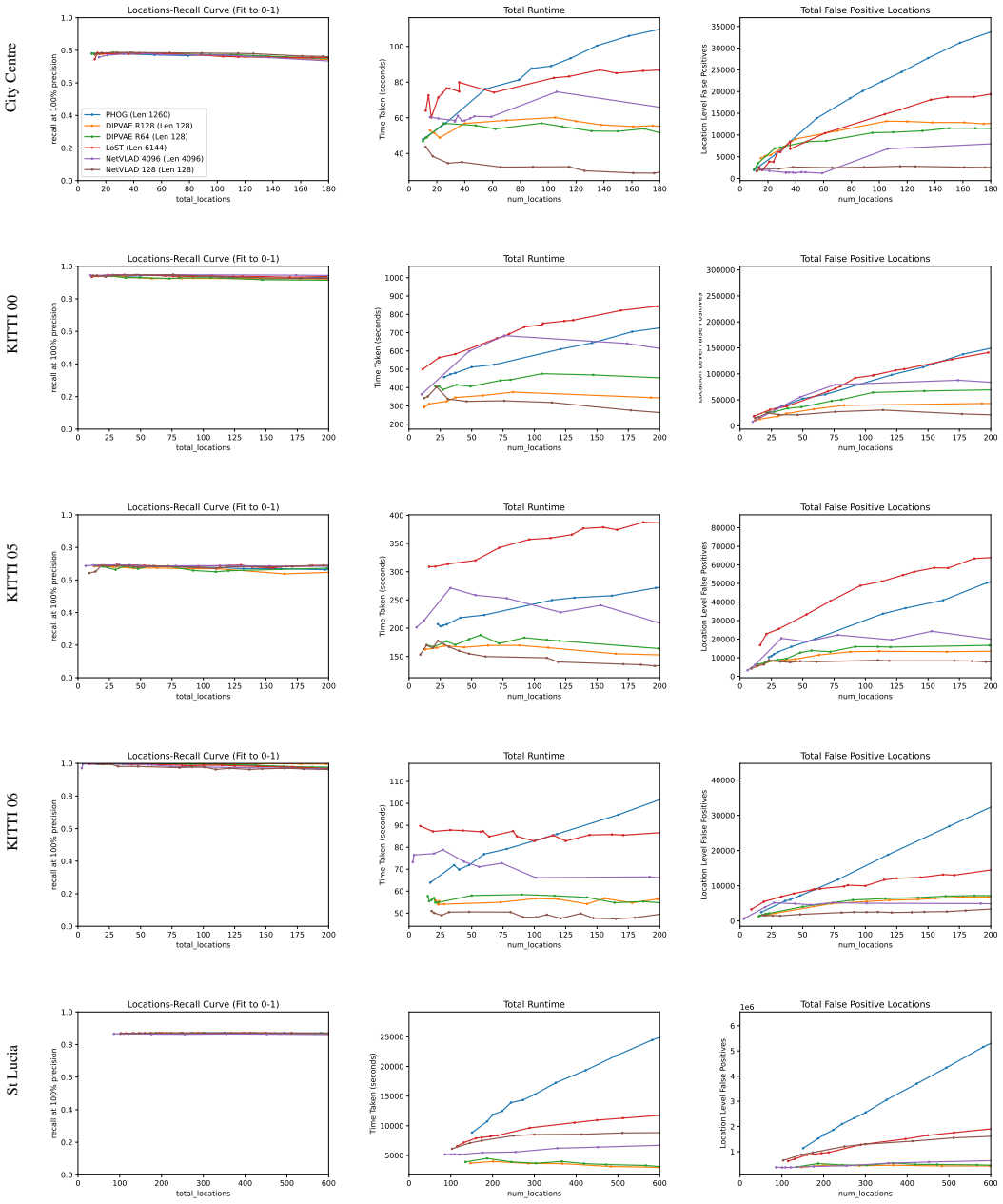


Figure 6.4: Results of evaluation of 6 global descriptors across 5 benchmark datasets. Each dot represents a single run, with the line showing a series of runs of the corresponding global descriptor on the respective dataset (shown on the left margin). A legend is given only in the first graph to avoid clutter. Column 1 highlights that DIPVAE (both R64 and R128) variants maintain the same performance, achieving recall values similar to those of other descriptors whilst being significantly more compact and faster to compute. Vector plots presented here are best viewed zoomed on a high-resolution screen.

t_{nn}	Recall	Locations	Image Loops
1.35	0.7772	24	436
1.40	0.7790	21	437
1.43	**0.7843**	**14**	**440**
1.50	**0.7790**	**16**	**437**
1.60	0.7451	12	418

Table 6.3: Table showing an example where a higher value of t_{nn} 1.50 yields more locations than that of t_{nn} 1.43, in rows 3 and 4 (highlighted in bold) respectively for LoST global descriptor on City Centre dataset

6.5 Discussion and Inference

The recall at 100% precision only changes negligibly among the evaluated candidates, as we can observe from Column 1 of Figure 6.4, showing that all global descriptors propose sufficient loop-closing location candidates without suffering any considerable drop in recall. However, the total time taken varies substantially and can be seen to be correlated in most of the cases with that of the total number of FPLC proposed. From the experimental results produced on different datasets, as the number of locations increases, the runtime and the number of FPLC proposed thereof increase linearly for PHOG, almost linearly for LoST (although less than that of PHOG), while NetVLAD and DIPVAE descriptors consistently show near-constant runtime with only a very small or negligible proportional increase in FPLC. Also, in both batch and single image descriptor compute times, DIPVAE gives the least compute time amongst all, followed by PHOG. On the other hand, NetVLAD and LoST require comparatively longer compute times, making them less suitable for realtime inference on autonomous vehicles.

Upon closer examination, we observed a significantly imbalanced distribution of images across locations in runs with longer runtimes. Maps with fewer locations comprising a very large number of images within locations diminish the advantage of the hierarchical approach and impair search efficiency, whereas an increased number of locations, each comprising only a very few images, leads to more location searches. Therefore, achieving optimal performance requires avoiding both overpopulated and underpopulated locations. The max intra-cluster distance t_{nn} affects the location cluster

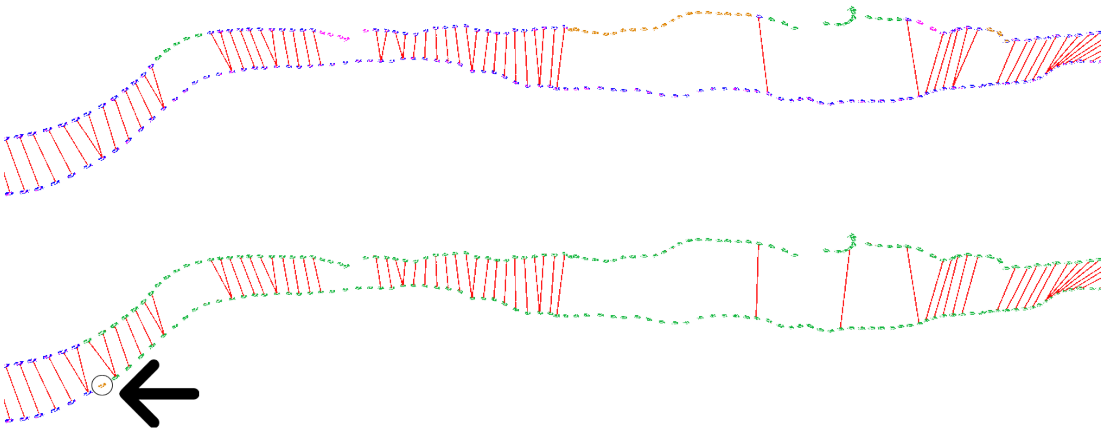


Figure 6.5: Screenshots of a section of the City Centre trajectory visualised in OdoViz [24] showing more fragmentation (different colours along the trajectory) for a higher value of t_{nn} 1.50 (top) compared to t_{nn} 1.43 (bottom). The car moves from left to right; time is represented in the z-axis with newer traversals presented higher up; poses that belong to the same location are in the same colour; and image loop closure proposals are shown in red. The missed loop closure proposal due to the creation of a new location (orange) in the bottom image is circled.

cohesion and separation, i.e., determines how many locations are formed and how populated the locations are. A small value for t_{nn} usually results in a map with a large number of total locations, and a larger t_{nn} value yields fewer locations.

However, this balance is also influenced by other factors. The location-image aggregation affects the loop closure proposals, and the loop closures affect the image aggregation dynamics for subsequent images, which in turn affect further loop closures. Consequently, it is possible that a slightly higher value of t_{nn} yields significantly more locations due to loop closures, as shown in Table 6.3. Furthermore, paradoxically, the generation of certain novel locations and subsequent loop closure misses can lead to an increase in overall recall in certain instances, which can be attributed to the dynamic hierarchical structure as shown in Figure 6.5.

Furthermore, our analysis reveals that the characteristics of the global descriptor has a very significant influence on the association of images with the locations. The generation of highly distinct descriptors for consecutive frames and images of physically proximal regions by a global descriptor results in a large number of locations, each containing only a few images. Although this may only lead to a negligible or slight

increase in FPLC, it still results in a significantly longer runtime (NetVLAD 4096 in Row 5, Column 2 and 3, in Figure 6.4) due to the sheer number of location searches triggered. We refer to this characteristic as continuity and posit that it is one of the significant factors in determining runtime. We determine continuity by computing the ratio of the number of locations containing fewer than t_{ci} images to the total number of locations, where t_{ci} is directly proportional to the frame rate of the camera and inversely proportional to the average speed of the car. Figure 6.6 presents a histogram depicting the number of images in all locations in St Lucia. As shown, for NetVLAD, due to its low continuity, the majority of locations are sparsely populated with fewer than 5 images, resulting in excessive fragmentation. Conversely, a global descriptor that produces similar global descriptors for images that are not physically nearby can lead to the unnecessary search of numerous known locations for place recognition, resulting in a large number of FPLC proposals and consequently increasing the runtime. We refer to this characteristic as distinctiveness, which is also crucial in determining runtime. We quantify distinctiveness by computing the inverse of the number of FPLC proposed by the global descriptor. The FPLC proposals for PHOG, as shown in Column 3 of Figure 6.4, increase significantly as the total number of locations increases, indicating its low distinctiveness.

Based on our empirical analysis, we hypothesise that an ideal global descriptor should possess the following characteristics:

- **Continuity**: Descriptor distance should gradually decrease as frames change continuously, resulting in smooth changes in similarity across space
- **Distinctiveness**: The descriptor distance between images from different regions should be significantly larger than the distance to its consecutive frames and images from similar regions.

To further substantiate this hypothesis, we conduct additional analysis utilising distance matrices (inverse similarity matrices) and t-SNE (t-distributed Stochastic Neighbour Embedding), which are presented in Figure 6.7. An example of a very smoothly changing but less distinctive nature exhibited by PHOG and an example of a less continuous but overly distinctive nature exhibited by NetVLAD are presented for the City Centre dataset. The distance matrices of the St Lucia dataset (21815×21815) are too large to interpret any meaningful information from and hence are not presented.

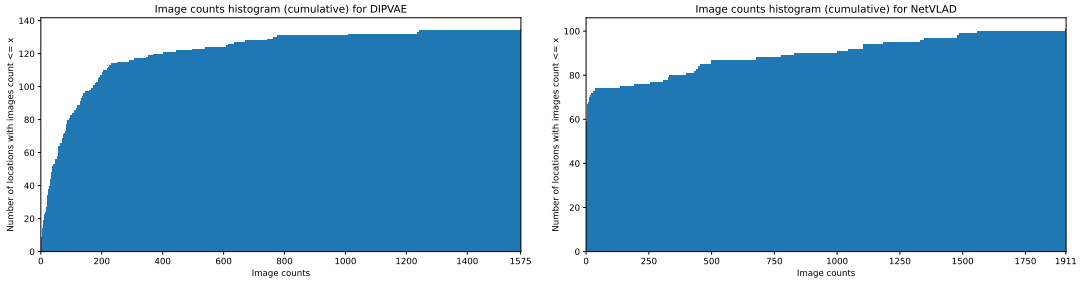


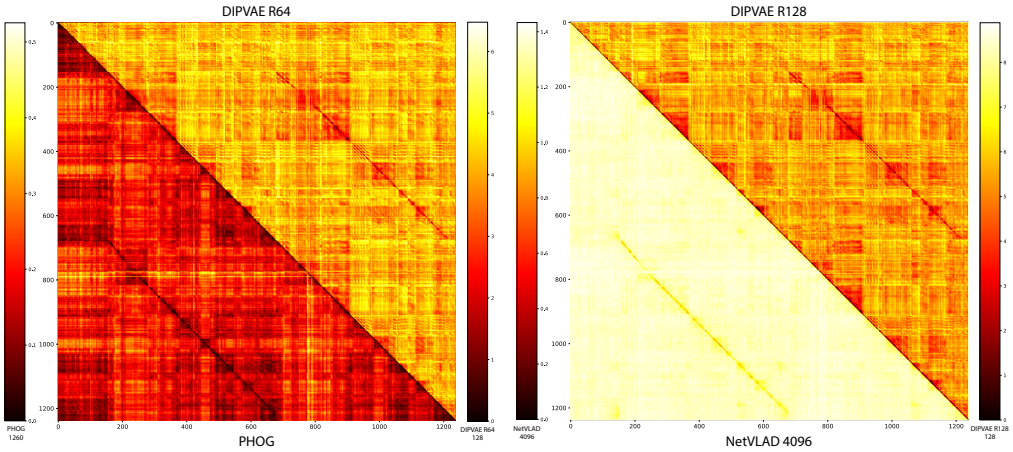
Figure 6.6: Cumulative histogram of image counts for the St Lucia dataset (21,815 images in total) for DIPVAE (left) and NetVLAD (right), illustrating the number of locations that possess an image count less than or equal to the given value of image count. Despite having a fewer total number of locations, NetVLAD has 65 out of 101 locations with 5 or fewer images, compared to 11 out of 135 for DIPVAE.

GDescriptor	**n(Loc)**	**Runtime (s) ↓**
PHOG	721	27939.47
LoST	654	12041.02
NetVLAD	624	6753.36
NetVLAD Cropped	698	8878.51
DIPVAE R128	665	**2967.95**
DIPVAE R64	609	3070.28

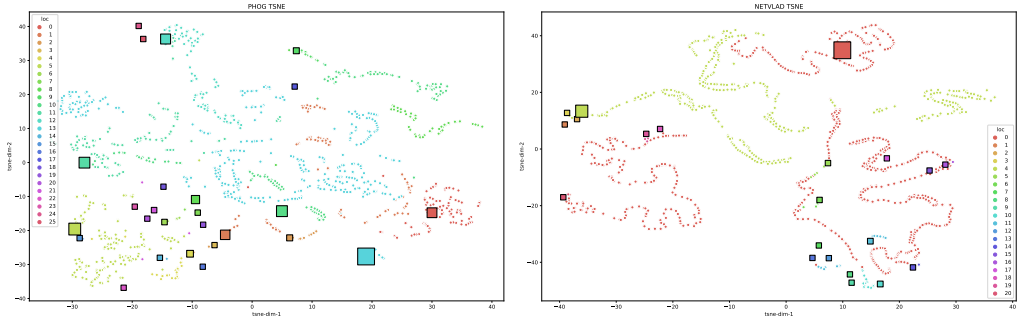
Table 6.4: Runtime on the St Lucia [46] dataset (17.6 km) for various descriptors, with the number of locations produced corresponding to that run.

PHOG exhibits an overabundance of similar descriptors and hence lacks distinctiveness, whereas NetVLAD has a scarcity of similar descriptors and hence lacks continuity. As such, the former leads to many false positive location matches, while the latter leads to poorly balanced locations (i.e., few locations having too many images and/or many locations having very few images), both resulting in decreased search efficiency and thus diminishing scalability. We also present the distance matrices of DIPVAE R64 and DIPVAE R128 for better comparison.

Columns 2 (total runtime) and 3 (total false positive locations) in Figure 6.4, particularly the last row corresponding to the longer St Lucia route, demonstrate the superiority of learned descriptors over handcrafted descriptors, which we argue is due to their inherent continuity and distinctiveness. Overall, both NetVLAD and DIPVAE show flat



(a) PHOG | DIPVAE R64 (b) NetVLAD | DIPVAE R128



(c) PHOG (d) NetVLAD

Figure 6.7: Distance matrices (top row) and t-SNE plots (bottom row) of different global descriptors for the City Centre dataset with 20-25 locations. To conserve space, we present the combined distance matrices (lower and upper triangular matrices) along with corresponding distance scales on the sides. In t-SNE plots, locations are shown in squares, with their size linked to the number of images they have, and their images are shown as dots of the same colour. Best viewed zoomed on a high-resolution screen.

runtimes and FPLC curves, supporting their usage as global descriptors and demonstrating their scalability in systems that deal with longer trajectories. However, we note that in the longest track evaluated, St Lucia (17.6km, 21.8k images), DIPVAE (both R128 and R64 variants) shows significantly lower runtimes, as shown in Table 6.4, while being very efficient with a compact embedding length of 128 and a substantially faster compute time, as noted in Table 6.2. We note that the LoST descriptor that encodes semantic information resulted in a longer runtime with more FPLC compared to its learned descriptor counterparts. We hypothesise that this is a result of the weaker discriminability arising due to ambiguous semantic information present across multiple regions.

6.6 Conclusion

We have extended HTMap [117], making improvements and incorporating learned global descriptors into the framework. We perform a comprehensive evaluation of various global descriptors on hierarchical topological mapping and present results of recall at 100% precision, total runtimes, and false positive loop closure location candidates (FPLC). All of the global descriptors we compared yield similar overall recall; however, they show crucial differences in runtimes. Based on our empirical analysis of multiple runs, we have identified that continuity and distinctiveness are crucial characteristics for an optimal global descriptor that enable efficient and scalable hierarchical mapping. Additionally, we have presented a methodology for quantifying and contrasting these characteristics. Our study demonstrates that the global descriptor based on an unsupervised learned Variational Autoencoder (VAE) excels in these characteristics and achieves significantly lower runtime. Consequently, on the longest track, St Lucia (17.6km), we observed that DIPVAE outperforms other global descriptors significantly, running up to 2.3x faster than the next best global descriptor, NetVLAD, and up to 9.5x faster than PHOG, while also being compact with an embedding length of 128. In the future, we intend to extend our analysis to more challenging datasets, including more extreme variation in weather, seasons, daylight, etc. We expect the performance of the learned descriptors to be more pronounced in these settings.

Chapter 7

Conclusion

7.1 Thesis Contributions

This thesis has explored the potential for learned image descriptors for Visual Place Recognition (VPR) systems, focussing on the development of approaches that enable scalable and efficient robotic perception pipelines. The thesis made a number of contributions in areas including robust learned image descriptors, scene categorisation, and hierarchical topological mapping and localisation. A principal motivation was addressing the challenges posed in VPR systems by variation in visual appearance as a result of changes in weather, seasons, and time of day, where the underlying approach was to leverage data-driven and deep learning techniques.

Our work began with a comprehensive exploration of VPR datasets, tools, and metrics, providing a solid foundation for the development and evaluation of VPR systems. We highlighted the importance of using diverse datasets that capture various environmental conditions for training robust VPR systems. Interpreting metadata, understanding dataset characteristics, and ensuring the relevance and comprehensiveness of training data, along with filtering and preparing subsets tailored to specific training requirements, arguably pose significant challenges for training learned descriptors. To address this, we presented OdoViz, a specialised research tool providing a unified framework for data visualisation, analysis, and processing. OdoViz facilitates working with large homogeneous datasets by offering a standardised framework, streamlining research workflows, and enabling efficient data handling. We emphasised the necessity

of curating tailored and standardised data subsets from public datasets for deep learning and further presented methods to generate them using OdoViz. With the discussion on metrics, we highlighted the importance of a comprehensive evaluation approach, emphasising the need to assess not only accuracy but also efficiency and scalability to ensure the system's practicality and robustness.

In the area of learned descriptors, training a VPR model on large sequential image datasets from public sources, which often exhibit long-term changes, presents significant challenges. These include redundant visual content in stationary scenes, such as intersections or roundabouts, and limited viewpoint variance on single carriageway roads, even with multiple traversals at different times. This hinders the training of robust learned descriptors using weakly supervised learning techniques such as contrastive loss and triplet loss, necessitating tailored data filtering and curation techniques, training regimes, and specialised loss functions. We addressed the challenge of training deep learning models with large sequential data to produce image representations that remain invariant to substantial environmental variations while retaining the ability to distinctly represent images from different locations. Central to achieving this was the novel approach of discretising trajectories into locations containing similar images from the same place to efficiently obtain unique triplets during training. We aggregated discretised locations combined with data augmentation techniques to add viewpoint variance. By discretising trajectories into locations and employing online triplet building strategies during training, we achieved significant time and computation savings. We modified the loss function and proposed architectural adaptations, which were key to enhanced training stability and retrieval performance of our learned representations in challenging conditions involving extreme day-night, weather, and seasonal changes.

Another significant challenge lies in understanding spatial configurations and object relationships to categorise the broader scene without explicitly identifying individual objects, focussed on deriving higher-level insights about the environment. In autonomous driving, scene categorisation into urban, rural, or suburban contexts provides crucial information for parametrising downstream tasks, such as pedestrian detection. One of the key challenges was designing a scene categorisation approach using a learned descriptor technique that is fast and efficient to compute, enabling it to be executed as a pretext task alongside computationally intensive tasks in the driving pipeline. Addressing this, we introduced a scalable and efficient novel unsupervised VAE-based approach for generating compact image representations. In particular, we designed

standard convolutional blocks with strided convolutions without the use of pooling and aggregation layers to allow more efficient, fast, low latency, and realtime inference using hardware convolutional accelerators. Unlike end-to-end models, our method leverages intermediate representations confined to a standard normal manifold, allowing easier comparison and interpretation. As such, the interpretability of these global descriptors, coupled with their compact nature (128-dimensional embeddings), presents significant advantages for real-time applications in autonomous vehicles. These descriptors, while central to scene categorisation, also hold potential for other tasks requiring high-level spatial reasoning.

As the operating environment of a robot scales, the computational complexity of exhaustively searching the database for potential matches increases linearly. Although faster retrieval methods and indexing techniques help mitigate this issue, the challenge becomes particularly pronounced when applied to mapping long trajectories spanning several kilometres. To address this, the reduction of search space complexity through hierarchical structures can be exploited to enhance the efficiency of VPR techniques on long trajectories. We built on top of the HTMap [117] algorithm and compared hierarchical topological mapping techniques with state-of-the-art handcrafted and learned global descriptors, presenting a comprehensive evaluation of their impact on performance. This aspect of the thesis proved particularly challenging, as it required decoupling individual components whose errors counteracted each other, ultimately resulting in similar overall recall performance. Our analysis revealed that, while various learned global descriptors achieved comparable overall recall, they exhibited significant differences in runtime and efficiency. Through this evaluation, we established that continuity and distinctiveness are critical characteristics of ideal global descriptors for hierarchical matching, supporting scalable and efficient visual localisation. To quantify and contrast these characteristics, we proposed a methodology to evaluate their effectiveness. Leveraging VAE-based learned global descriptors that capture high-level scene information, we demonstrated efficient and scalable hierarchical topological mapping, achieving superior runtime performance, particularly on long trajectories. This, in turn, contributes to the overarching aim of this thesis: to leverage learned descriptors to enable their efficient and scalable deployment in real-world navigation applications over large-scale environments.

Prior to this work, limitations in utilising large sequential datasets, achieving efficient scene categorisation, and enabling scalable hierarchical mapping posed significant

obstacles to adopting learned descriptors for robust, efficient, and fast operations. Taken as a whole, the contributions of this thesis provide a strong foundation for future research and development of learned VPR techniques, bringing VPR research one step closer towards realising highly efficient and scalable systems capable of operating in diverse and challenging real-world environments.

7.2 Opportunities for Future Work

Building upon the foundations laid in this thesis, several avenues for future research emerge:

Multi-modal backbones: Transformer models have demonstrated significant success in various domains due to their ability to handle sequential data and capture long-range dependencies, as seen in vision-language tasks such as CLIP [237] and Blip [238, 239]. Such an architecture would potentially allow the integration of textual information extracted from visual data alongside global image descriptors, potentially enhancing the system's understanding of the context of the image presented. By leveraging the attention mechanisms inherent in transformers and sharing the knowledge learned from textual descriptions of images, the system could learn to focus on the most relevant features. Recent advancements in self-supervised learning, such as MoCo [101, 102, 103], SimCLR [99, 100], DINO [105] and MAE [240], have shown the potential of transformer-based architectures in generating high-quality representations without relying on large labelled datasets. These methods, primarily applied to object-focussed images, utilise contrastive learning or masked prediction tasks to learn robust visual features. Extending these techniques to more complex scenes, encompassing a large field of view with multiple objects, could provide significant benefits for scene understanding tasks. Although transformers are highly effective for multi-modal learning, their computational complexity remains a challenge, especially in resource-constrained environments such as embedded systems. Investigating compute-efficient approaches [241, 242, 243, 244] could allow for the deployment of systems employing transformer backbones.

Extended Scene Categorisation: Exploring the addition of categories such as motorways, tunnels, and car parks — alongside the existing categories, rural, urban,

and suburban — would provide richer context for autonomous driving tasks, enabling more precise and context-aware decision-making. These new categories could be useful for applications like route planning, where understanding the specific driving environment is essential for adjusting driving behaviour and improving safety [245, 246]. For instance, recognising a motorway would allow the system to adapt speed, lane positioning, and traffic monitoring accordingly. Similarly, recognising tunnels and car parks could trigger specific navigation strategies, such as initiating low-speed manoeuvres in confined spaces and enabling GPS-independent actions like mapping and localisation. Moreover, incorporating temporal consistency into the categorisation process, while broadening the range of scene categories, could potentially improve categorisation accuracy, ensuring smoother and more accurate transitions between scene categories [247, 248]. Autonomous driving tasks often require continuous real-time scene analysis, where successive frames of video data contain highly correlated information [249]. For example, the system could avoid misclassifying short-term visual ambiguities, such as shadows or reflections, by considering the consistency of the scene over a period of time. Integrating auxiliary data sources like image quality metrics (brightness, contrast, and sharpness values used by the sensor), vehicle speed, GPS, and heading could further improve its accuracy, allowing the model to account for additional contextual clues, which is essential for accurate scene categorisation. Additionally, other vision tasks requiring analysis of individual objects — for example, performing semantic segmentation to extract vegetation or detecting specific objects and landmarks for the purpose of localisation — can feed back into the broader scene categorisation task. This integration of diverse scene categories, temporal consistency, and additional context would ultimately contribute to a more comprehensive understanding of the driving environment, supporting safer and more efficient autonomous navigation.

Photorealistic Synthetic Training Data: The use of synthetic data for training deep learning models for VPR tasks has been limited largely due to a significant gap between real and synthetic datasets. Models trained on synthetic data often fail to generalise well when directly applied to real-world scenarios without the use of domain adaptation and fine-tuning techniques, as synthetic data struggles to replicate the nuances and variability of real-world images [158, 250, 251, 252]. However, recent advancements in rendering capabilities and generative techniques, particularly using diffusion models and GANs, have allowed for the creation of photorealistic synthetic images that closely mimic real objects, narrowing this gap significantly [209, 158]. Despite these advancements,

challenges remain for generating photorealistic synthetic data for complex scenes with multiple objects in a large field of view and varied environmental factors. Unlike object-focussed images, which are relatively straightforward for current generative models, full scenes require the network to handle diverse object interactions, depth variations, and context continuity across large spaces. By synthesising varied scenarios, including rare or hazardous situations, synthetic data can expand the range of training samples far beyond what is feasible with real-world data alone. Further research into scene-focussed synthetic data, particularly with advanced generative models fine-tuned for scene realism, could enable models to perform well in complex environments without extensive real-world training datasets [253, 158].

Fully Learned Descriptors for Hierarchical Mapping and Localisation: The use of learned image descriptors for both global and local representations, moving beyond traditional point-based features for more efficient mapping, could allow for a more efficient image matching process. While the current pipeline requires the use of keypoint-based descriptors for geometric verification using epipolar checks to avoid false positives, this reliance on traditional feature descriptors presents a bottleneck. In a learned descriptor framework, detection and description are integrated within a single model, capitalising on GPU parallelism. However, shifting directly to learned descriptors without the geometric verification step could increase the risk of false positives, which can lead to catastrophic failures in mapping and localisation applications. Recent works in learning-based localisation, such as SuperGlue [254], DELG [255], DFM [256], D2-Net [16], and LF-Net [257], have introduced learned matching techniques that partially integrate geometric reasoning through deep neural networks. However, achieving absolute reliability remains a critical challenge, especially in mapping applications requiring 100% precision. Future work could focus on enhancing these learning-based methods by more deeply embedding geometric reasoning into the learning process and directly incorporating geometric constraints into the loss function. Another promising direction is the development of hierarchical descriptors that integrate both global and local context within a single representation, enabling coarse-to-fine matching without requiring separate descriptors at each hierarchical level. This approach streamlines computation by avoiding multiple descriptor feature extraction processes, enhancing efficiency in mapping and localisation tasks.

Model Compression and Optimisation for Edge Deployment: For deploying the approaches described in the thesis in real-time autonomous systems, it is crucial to opti-

mise performance on edge hardware, such as hardware convolutional accelerators and automotive-grade chips. These devices are often constrained by limited computational resources and power efficiency requirements, making neural network optimisation essential for maintaining system performance [258]. One of the key strategies for achieving this is neural network pruning, where unimportant or redundant connections within the neural network model are identified and removed, leading to a significant reduction in model complexity. Structured pruning techniques not only decrease the model size but also accelerate inference times, which is critical for realtime tasks in edge devices [259, 260]. Quantization is another key technique to explore for optimising neural networks for edge deployment. By reducing the precision of the weights and activations, typically from 32-bit floating point to lower-precision 8-bit floats or integers, quantization can greatly reduce the memory footprint and computational requirements of the model [261, 262, 263]. Furthermore, Quantization-Aware Training (QAT) could be employed during the training phase to mitigate any performance degradation that occurs from lower precision representation, allowing for more robust model performance on resource-constrained hardware [264, 265]. Exploring hardware-specific optimisations through frameworks such as TensorRT and Apache TVM can further improve performance, enabling low-latency and energy-efficient deployment.

Adaptive inference strategies allow the VPR system to dynamically adjust its computational load and accuracy based on realtime conditions such as vehicle speed, scene complexity, and hardware limitations [266, 267]. For example, the system could shift between lightweight, faster models during simpler scenarios such as highway driving and more complex, accurate models in dense environments, like urban areas. One prospective approach within adaptive inference is the use of early-exit architectures, where intermediate outputs are assessed at multiple points within the network [268, 269, 270]. These architectures allow the model to terminate inference earlier when sufficient confidence in the output is achieved, reducing computation time significantly. Early-exit networks have shown efficacy in achieving low-latency, high-accuracy results in real-time applications by leveraging the early layers for less complex inputs and fully utilising the network only when necessary. Dynamic pruning is another key technique for adaptive inference. In this approach, certain network layers or neurons are selectively activated or deactivated based on the specific input, thereby optimising computational efficiency without degrading accuracy [271, 272, 273]. Dynamic pruning is thus potentially useful in tasks where input complexity varies significantly, as it

enables temporary resource allocation adjustments based on current processing needs. By incorporating such adaptive inference methods, a VPR system could intelligently balance computational demands with accuracy, tailoring inference to both environmental and hardware constraints, ultimately improving performance, power efficiency, and reliability in real-world driving conditions.

Bibliography

[1] S. Schubert, P. Neubert, S. Garg, M. Milford, and T. Fischer, "Visual Place Recognition: A Tutorial," *IEEE Robotics and Automation Magazine*, pp. 2–16, 2023.

[2] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2015.

[3] Norwegian Broadcasting Corporation, "Nordlandsbanen – minutt for minutt." https://tv.nrk.no/serie/nordlandsbanen-minutt-for-minutt, 2012.

[4] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[5] X. Zhang, L. Wang, and Y. Su, "Visual place recognition: A survey from deep learning perspective," *Pattern Recognition*, vol. 113, p. 107760, 2021.

[6] E. Olson and P. Agarwal, "Inference on networks of mixtures for robust robot mapping," *Int. J. Rob. Res.*, vol. 32, p. 826–840, jun 2013.

[7] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 Year, 1000km: The Oxford RobotCar Dataset," *The International Journal of Robotics Research (IJRR)*, vol. 36, no. 1, pp. 3–15, 2017.

[8] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int. J. Comput. Vision*, vol. 60, pp. 91–110, Nov. 2004.

[9] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Comput. Vis. Image Underst.*, vol. 110, pp. 346–359, June 2008.

[10] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Computer Vision – ECCV 2010* (K. Daniilidis, P. Maragos, and N. Paragios, eds.), (Berlin, Heidelberg), pp. 778–792, Springer Berlin Heidelberg, 2010.

[11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*, pp. 2564–2571, 2011.

[12] P. F. Alcantarilla, J. Nuevo, and A. Bartoli, "Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces," in *British Machine Vision Conf. (BMVC)*, 2013.

[13] Sivic and Zisserman, "Video Google: A Text Retrieval Approach to Object Matching in Videos," in *Proceedings Ninth IEEE International Conference on Computer Vision*, pp. 1470–1477 vol.2, Oct 2003.

[14] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Workshop on statistical learning in computer vision, ECCV*, vol. 1, Prague, 2004.

[15] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman, "Discovering objects and their location in images," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 1, pp. 370–377, IEEE, 2005.

[16] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *2010 IEEE computer society conference on computer vision and pattern recognition*, pp. 3304–3311, IEEE, 2010.

[17] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, IEEE, 2007.

[18] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 886–893 vol. 1, 2005.

[19] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International journal of computer vision*, vol. 42, no. 3, pp. 145–175, 2001.

[20] T. Naseer, G. L. Oliveira, T. Brox, and W. Burgard, "Semantics-aware visual localization under challenging perceptual conditions," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2614–2620, IEEE, 2017.

[21] S. R. Dubey, "A Decade Survey of Content Based Image Retrieval Using Deep Learning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 5, pp. 2687–2704, 2022.

[22] R. S. Sutton, "The bitter lesson." http://www.incompleteideas.net/IncIdeas/BitterLesson.html, March 2019.

[23] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *2006 IEEE CVPR'06*, vol. 2, pp. 2161–2168, Ieee, 2006.

[24] S. Ramachandran and J. McDonald, "OdoViz: A 3D Odometry Visualization and Processing Tool," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 1391–1398, IEEE, 2021.

[25] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. USA: McGraw-Hill, Inc., 1986.

[26] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary Robust Invariant Scalable Keypoints," in *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, (Washington, DC, USA), pp. 2548–2555, IEEE Computer Society, 2011.

[27] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast Retina Keypoint," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 510–517, 2012.

[28] X. Yang and K.-T. T. Cheng, "Local Difference Binary for Ultrafast and Distinctive Feature Description," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, pp. 188–194, 2014.

[29] D. Filliat, "A visual bag of words method for interactive qualitative localization and mapping," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 3921–3926, 2007.

[30] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer, "Fast and Incremental Method for Loop-Closure Detection Using Bags of Visual Words," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1027–1037, 2008.

[31] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat, "Incremental vision-based topological SLAM," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1031–1036, 2008.

[32] G. Qiu, "Indexing chromatic and achromatic patterns for content-based colour image retrieval," *Pattern Recognition*, vol. 35, no. 8, pp. 1675–1686, 2002.

[33] Y.-G. Jiang, C.-W. Ngo, and J. Yang, "Towards optimal bag-of-features for object categorization and semantic video retrieval," in *Proceedings of the 6th ACM international conference on Image and video retrieval*, pp. 494–501, 2007.

[34] L. Ai, J. Yu, and T. Guan, "Spherical soft assignment: Improving image representation in content-based image retrieval," in *Advances in Multimedia Information Processing – PCM 2012* (W. Lin, D. Xu, A. Ho, J. Wu, Y. He, J. Cai, M. Kankanhalli, and M.-T. Sun, eds.), (Berlin, Heidelberg), pp. 801–810, Springer Berlin Heidelberg, 2012.

[35] R. Arandjelovic and A. Zisserman, "All about VLAD," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1578–1585, 2013.

[36] A. Vedaldi and B. Fulkerson, "VLFeat: An Open and Portable Library of Computer Vision Algorithms." https://www.vlfeat.org/api/fisher-fundamentals.html, 2008.

[37] T. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers," in *Advances in neural information processing systems*, pp. 487–493, 1999.

[38] M. Seeland, M. Rzanny, N. Alaqraa, J. Wäldchen, and P. Mäder, "Plant species classification using flower images—A comparative study of local feature representations," *PloS one*, vol. 12, no. 2, 2017.

[39] M. Cummins and P. Newman, "Appearance-only SLAM at Large Scale with FAB-MAP 2.0," *Int. J. Rob. Res.*, vol. 30, pp. 1100–1123, Aug. 2011.

[40] D. Gálvez-López and J. D. Tardós, "Bags of Binary Words for Fast Place Recognition in Image Sequences," *IEEE Transactions on Robotics*, vol. 28, pp. 1188–1197, October 2012.

[41] W. Maddern, A. Stewart, C. McManus, B. Upcroft, W. Churchill, and P. Newman, "Illumination invariant imaging: Applications in robust vision-based localisation, mapping and classification for autonomous vehicles," in *Proceedings of the Visual Place Recognition in Changing Environments Workshop, IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China*, vol. 2, p. 3, 2014.

[42] R. Arroyo, P. F. Alcantarilla, L. M. Bergasa, and E. Romera, "Towards life-long visual localization using an efficient matching of binary sequences from images," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6328–6335, 2015.

[43] H. Badino, D. Huber, and T. Kanade, "Real-time topometric localization," in *2012 IEEE International Conference on Robotics and Automation*, pp. 1635–1642, IEEE, 2012.

[44] N. Sünderhauf and P. Protzel, "Brief-gist-closing the loop by simple means," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1234–1241, IEEE, 2011.

[45] M. J. Milford and G. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights," in *2012 IEEE International Conference on Robotics and Automation*, pp. 1643–1649, 2012.

[46] A. J. Glover, W. P. Maddern, M. J. Milford, and G. F. Wyeth, "FAB-MAP+ RatSLAM: Appearance-based SLAM for multiple times of day," in *2010 IEEE international conference on robotics and automation*, pp. 3507–3512, IEEE, 2010.

[47] N. Sünderhauf, P. Neubert, and P. Protzel, "Are we there yet? Challenging SeqSLAM on a 3000 km journey across all four seasons," in *Proc. of Workshop on Long-Term Autonomy, IEEE International Conference on Robotics and Automation (ICRA)*, p. 2013, 2013.

[48] I. Ulrich and I. Nourbakhsh, "Appearance-based place recognition for topological localization," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, pp. 1023–1029 vol.2, 2000.

[49] J. Kosecka, L. Zhou, P. Barber, and Z. Duric, "Qualitative image based localization in indoors environments," in *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, vol. 2, pp. II–II, 2003.

[50] A. Bosch, A. Zisserman, and X. Munoz, "Representing Shape with a Spatial Pyramid Kernel," in *Proceedings of the 6th ACM CIVR 2007*, (New York, NY, USA), p. 401–408, ACM, 2007.

[51] N. Nourani-Vatani, P. V. Borges, J. M. Roberts, and M. V. Srinivasan, "On the Use of Optical Flow for Scene Change Detection and Description," *J. Intell. Robotics Syst.*, vol. 74, p. 817–846, jun 2014.

[52] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in neural information processing systems*, pp. 396–404, 1990.

[53] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

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

[55] F. Yu and V. Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2016.

[56] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii, and T. Sattler, "D2-Net: A Trainable CNN for Joint Description and Detection of Local Features," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8084–8093, 2019.

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

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[59] E. Xing, M. Jordan, S. J. Russell, and A. Ng, "Distance Metric Learning with Application to Clustering with Side-Information," in *Advances in Neural Information Processing Systems* (S. Becker, S. Thrun, and K. Obermayer, eds.), vol. 15, MIT Press, 2002.

[60] L. Yang and R. Jin, "Distance metric learning: A comprehensive survey," *Michigan State Universiy*, vol. 2, no. 2, p. 4, 2006.

[61] P. Moutafis, M. Leng, and I. A. Kakadiaris, "An Overview and Empirical Comparison of Distance Metric Learning Methods," *IEEE Transactions on Cybernetics*, vol. 47, no. 3, pp. 612–625, 2017.

[62] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality Reduction by Learning an Invariant Mapping," in *2006 IEEE CVPR*, vol. 2, pp. 1735–1742, June 2006.

[63] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," *2015 IEEE CVPR*, Jun 2015.

[64] W. Chen, X. Chen, J. Zhang, and K. Huang, "Beyond triplet loss: a deep quadruplet network for person re-identification," *CoRR*, vol. abs/1704.01719, 2017.

[65] A. Hermans, L. Beyer, and B. Leibe, "In Defense of the Triplet Loss for Person Re-Identification," 2017.

[66] D. Kim and M. R. Walter, "Satellite image-based localization via learned embeddings," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2073–2080, May 2017.

[67] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN Architecture for Weakly Supervised Place Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 6, pp. 1437–1451, 2018.

[68] J. Yu, C. Zhu, J. Zhang, Q. Huang, and D. Tao, "Spatial Pyramid-Enhanced NetVLAD With Weighted Triplet Loss for Place Recognition," *IEEE TNNLS*, vol. 31, no. 2, pp. 661–674, 2020.

[69] S. Hausler, S. Garg, M. Xu, M. Milford, and T. Fischer, "Patch-NetVLAD: Multi-Scale Fusion of Locally-Global Descriptors for Place Recognition," in *IEEE/CVF CVPR*, pp. 14136–14147, 2021.

[70] Y. Cai, J. Zhao, J. Cui, F. Zhang, T. Feng, and C. Ye, "Patch-NetVLAD+: Learned patch descriptor and weighted matching strategy for place recognition," in *2022 IEEE MFI*, pp. 1–8, 2022.

[71] A. Khaliq, M. Milford, and S. Garg, "MultiRes-NetVLAD: Augmenting Place Recognition Training With Low-Resolution Imagery," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3882–3889, 2022.

[72] Y. Zhang, Q. Zhong, L. Ma, D. Xie, and S. Pu, "Learning incremental triplet margin for person re-identification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 9243–9250, 2019.

[73] J. Lu, C. Xu, W. Zhang, L.-Y. Duan, and T. Mei, "Sampling Wisely: Deep Image Embedding by Top-k Precision Optimization," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7961–7970, 2019.

[74] Y. Ge, H. Wang, F. Zhu, R. Zhao, and H. Li, "Self-supervising Fine-grained Region Similarities for Large-scale Image Localization," in *European Conference on Computer Vision*, 2020.

[75] L. Liu, H. Li, and Y. Dai, "Stochastic Attraction-Repulsion Embedding for Large Scale Image Localization," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2570–2579, 2019.

[76] G. Peng, Y. Yue, J. Zhang, Z. Wu, X. Tang, and D. Wang, "Semantic Reinforced Attention Learning for Visual Place Recognition," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13415–13422, 2021.

[77] G. Berton, C. Masone, and B. Caputo, "Rethinking Visual Geo-Localization for Large-Scale Applications," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4878–4888, June 2022.

[78] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "CosFace: Large Margin Cosine Loss for Deep Face Recognition," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5265–5274, 2018.

[79] F. Radenović, G. Tolias, and O. Chum, "Fine-Tuning CNN Image Retrieval with No Human Annotation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1655–1668, 2019.

[80] Y. Xu, Y. Han, R. Hong, and Q. Tian, "Sequential Video VLAD: Training the Aggregation Locally and Temporally," *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 4933–4944, 2018.

[81] L. Wu, Y. Wang, L. Shao, and M. Wang, "3-D PersonVLAD: Learning Deep Global Representations for Video-Based Person Reidentification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3347–3359, 2019.

[82] T. Naseer, W. Burgard, and C. Stachniss, "Robust Visual Localization Across Seasons," *IEEE Transactions on Robotics*, vol. 34, no. 2, pp. 289–302, 2018.

[83] O. Vysotska and C. Stachniss, "Effective visual place recognition using multi-sequence maps," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1730–1736, 2019.

[84] R. Dube, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, "SegMatch: Segment based place recognition in 3D point clouds," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5266–5272, May 2017.

[85] S. Garg, N. Suenderhauf, and M. Milford, "Lost? appearance-invariant place recognition for opposite viewpoints using visual semantics," *arXiv preprint arXiv:1804.05526*, 2018.

[86] G. Lin, A. Milan, C. Shen, and I. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1925–1934, 2017.

[87] G. E. Hinton and R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[88] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *CoRR*, vol. abs/1312.6114, 2013.

[89] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in *Advances in*

Neural Information Processing Systems (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, eds.), vol. 27, Curran Associates, Inc., 2014.

[90] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A Survey on Contrastive Self-Supervised Learning," *Technologies*, vol. 9, no. 1, 2021.

[91] J. Gui, T. Chen, J. Zhang, Q. Cao, Z. Sun, H. Luo, and D. Tao, "A Survey on Self-Supervised Learning: Algorithms, Applications, and Future Trends," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 9052–9071, 2024.

[92] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *European conference on computer vision*, pp. 649–666, Springer, 2016.

[93] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros, "Context Encoders: Feature Learning by Inpainting," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2536–2544, 2016.

[94] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised Representation Learning by Predicting Image Rotations," in *International Conference on Learning Representations*, 2018.

[95] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised Visual Representation Learning by Context Prediction," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, (USA), p. 1422–1430, IEEE Computer Society, 2015.

[96] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE CVPR*, pp. 248–255, 2009.

[97] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision – ECCV 2014* (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), (Cham), pp. 740–755, Springer International Publishing, 2014.

[98] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[99] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*, pp. 1597–1607, PMLR, 2020.

[100] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, "Big self-supervised models are strong semi-supervised learners," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, (Red Hook, NY, USA), Curran Associates Inc., 2020.

[101] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum Contrast for Unsupervised Visual Representation Learning," in *2020 IEEE/CVF CVPR*, pp. 9726–9735, 2020.

[102] X. Chen, H. Fan, R. Girshick, and K. He, "Improved Baselines with Momentum Contrastive Learning," *arXiv preprint arXiv:2003.04297*, 2020.

[103] X. Chen, S. Xie, and K. He, "An Empirical Study of Training Self-Supervised Vision Transformers," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9620–9629, 2021.

[104] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko, "Bootstrap Your Own Latent: A New Approach to Self-Supervised Learning," 2020.

[105] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, R. Howes, P.-Y. Huang, H. Xu, V. Sharma, S.-W. Li, W. Galuba, M. Rabbat, M. Assran, N. Ballas, G. Synnaeve, I. Misra, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, "DINOv2: Learning Robust Visual Features without Supervision," 2023.

[106] N. Keetha, A. Mishra, J. Karhade, K. M. Jatavallabhula, S. Scherer, M. Krishna, and S. Garg, "AnyLoc: Towards Universal Visual Place Recognition," *IEEE Robotics and Automation Letters*, vol. 9, no. 2, pp. 1286–1293, 2023.

[107] T. Sattler, B. Leibe, and L. Kobbelt, "Efficient & effective prioritized matching for large-scale image-based localization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 9, pp. 1744–1756, 2016.

[108] L. Svärm, O. Enqvist, F. Kahl, and M. Oskarsson, "City-scale localization for cameras with known vertical direction," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1455–1461, 2016.

[109] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From Coarse to Fine: Robust Hierarchical Localization at Large Scale," in *2019 IEEE/CVF CVPR*, pp. 12708–12717, 2019.

[110] M. Cummins and P. Newman, "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[111] H. Badino, D. Huber, and T. Kanade, "Visual topometric localization," in *2011 IEEE Intelligent vehicles symposium (IV)*, pp. 794–799, IEEE, 2011.

[112] J. Lim, J.-M. Frahm, and M. Pollefeys, "Online environment mapping using metric-topological maps," *The International Journal of Robotics Research*, vol. 31, no. 12, pp. 1394–1408, 2012.

[113] M. Dymczyk, S. Lynen, M. Bosse, and R. Siegwart, "Keep it brief: Scalable creation of compressed localization maps," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2536–2542, IEEE, 2015.

[114] F. Blochliger, M. Fehr, M. Dymczyk, T. Schneider, and R. Siegwart, "Topomap: Topological mapping and navigation based on visual slam maps," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3818–3825, IEEE, 2018.

[115] G. L. Oliveira, N. Radwan, W. Burgard, and T. Brox, "Topometric localization with deep learning," in *Robotics Research*, pp. 505–520, Springer, 2020.

[116] D. S. Chaplot, R. Salakhutdinov, A. Gupta, and S. Gupta, "Neural topological slam for visual navigation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12875–12884, 2020.

[117] E. Garcia-Fidalgo and A. Ortiz, "Hierarchical Place Recognition for Topological Mapping," *IEEE Transactions on Robotics*, vol. 33, pp. 1061–1074, Oct 2017.

[118] E. D. Dickmanns, *Dynamic Vision for Perception and Control of Motion*. Springer London, 2007.

[119] J. Delcker, "The man who invented the self-driving car (in 1986)." https://www.politico.eu/article/delf-driving-car-born-1986-ernst-dickmanns-mercedes/.

[120] J. Schmidhuber, "Highlights of Robot Cars History." https://people.idsia.ch//~juergen/robotcars.html.

[121] D. Scharstein, R. Szeliski, and R. Zabih, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," in *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*, pp. 131–140, 2001.

[122] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "SUN database: Large-scale scene recognition from abbey to zoo," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3485–3492, 2010.

[123] M. Everingham, S. M. Eslami, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes Challenge: A Retrospective," *Int. J. Comput. Vision*, vol. 111, p. 98–136, jan 2015.

[124] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE transactions on PAMI*, vol. 40, no. 6, pp. 1452–1464, 2017.

[125] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene Parsing through ADE20K Dataset," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5122–5130, 2017.

[126] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on CVPR*, pp. 3354–3361, IEEE, 2012.

[127] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, F. Kahl, and T. Pajdla, "Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[128] T. Naseer, L. Spinello, W. Burgard, and C. Stachniss, "Robust Visual Robot Localization Across Seasons Using Network Flows," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pp. 2564–2570, AAAI Press, 2014.

[129] A. Torii, J. Sivic, T. Pajdla, and M. Okutomi, "Visual Place Recognition with Repetitive Structures," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[130] A. Torii, J. Sivic, M. Okutomi, and T. Pajdla, "Visual Place Recognition with Repetitive Structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 11, pp. 2346–2359, 2015.

[131] A. Torii, R. Arandjelović, J. Sivic, M. Okutomi, and T. Pajdla, "24/7 place recognition by view synthesis," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1808–1817, 2015.

[132] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3213–3223, 2016.

[133] N. Gählert, N. Jourdan, M. Cordts, U. Franke, and J. Denzler, "Cityscapes 3D: Dataset and Benchmark for 9 DoF Vehicle Detection," 2020.

[134] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning," 2018.

[135] D. M. Chen, G. Baatz, K. Köser, S. S. Tsai, R. Vedantham, T. Pylvänäinen, K. Roimela, X. Chen, J. Bach, M. Pollefeys, B. Girod, and R. Grzeszczuk, "City-scale landmark identification on mobile devices," in *CVPR 2011*, pp. 737–744, 2011.

[136] M. Warren, D. McKinnon, H. He, and B. Upcroft, "Unaided Stereo Vision Based Pose Estimation," in *Australasian Conference on Robotics and Automation*, (Brisbane), 2010.

[137] G. Neuhold, T. Ollmann, S. R. Bulò, and P. Kontschieder, "The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 5000–5009, 2017.

[138] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt, "Image Retrieval for Image-Based Localization Revisited," in *British Machine Vision Conference*, 2012.

[139] V. Balntas, D. Frost, R. Kouskouridas, A. Barroso-Laguna, A. Talattof, H. Heijnen, and K. Mikolajczyk, "Scape Imperial Localization Dataset." https://research.scape.io/silda, 2020.

[140] S. Griffith, G. Chahine, and C. Pradalier, "Symphony Lake Dataset," *The International Journal of Robotics Research*, vol. 36, no. 11, pp. 1151–1158, 2017.

[141] C. Toft, W. Maddern, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, T. Pajdla, *et al.*, "Long-term visual localization revisited," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 4, pp. 2074–2088, 2020.

[142] H. Taira, M. Okutomi, T. Sattler, M. Cimpoi, M. Pollefeys, J. Sivic, T. Pajdla, and A. Torii, "InLoc: Indoor visual localization with dense matching and view synthesis," in *CVPR*, 2018.

[143] ETH Zurich Computer Vision Group and Microsoft Mixed Reality & AI Lab Zurich, "The ETH-Microsoft Localization Dataset." https://github.com/cvg/visloc-iccv2021, 2021.

[144] M. Humenberger, Y. Cabon, N. Guérin, J. Morat, J. Revaud, P. Rerole, N. Pion, C. R. de Souza, V. Leroy, and G. Csurka, "Robust Image Retrieval-based Visual Localization using Kapture," *ArXiv*, vol. abs/2007.13867, 2020.

[145] M. Måns Larsson, E. Stenborg, L. Hammarstrand, M. Pollefeys, T. Sattler, and F. Kahl, "A Cross-Season Correspondence Dataset for Robust Semantic Segmentation," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9524–9534, 2019.

[146] S. Yogamani, C. Hughes, J. Horgan, G. Sistu, P. Varley, D. O'Dea, M. Uricar, S. Milz, M. Simon, K. Amende, C. Witt, H. Rashed, S. Chennupati, S. Nayak,

S. Mansoor, X. Perrotton, and P. Perez, "WoodScape: A Multi-Task, Multi-Camera Fisheye Dataset for Autonomous Driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[147] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A Multimodal Dataset for Autonomous Driving," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11618–11628, 2020.

[148] F. Warburg, S. Hauberg, M. López-Antequera, P. Gargallo, Y. Kuang, and J. Civera, "Mapillary Street-Level Sequences: A Dataset for Lifelong Place Recognition," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2623–2632, 2020.

[149] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A Novel Dataset and Benchmarks for Urban Scene Understanding in 2D and 3D," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 3, pp. 3292–3310, 2023.

[150] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[151] D. Hernandez-Juarez, L. Schneider, A. Espinosa, D. Vazquez, A. M. Lopez, U. Franke, M. Pollefeys, and J. C. Moure, "Slanted Stixels: Representing San Francisco's Steepest Streets," in *British Machine Vision Conference (BMVC), 2017*, 2017.

[152] J. Zolfaghari Bengar, A. Gonzalez-Garcia, G. Villalonga, B. Raducanu, H. H. Aghdam, M. Mozerov, A. M. Lopez, and J. van de Weijer, "Temporal Coherence for Active Learning in Videos," *arXiv preprint arXiv:1908.11757*, 2019.

[153] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "VirtualWorlds as Proxy for Multi-object Tracking Analysis," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4340–4349, 2016.

[154] Y. Cabon, N. Murray, and M. Humenberger, "Virtual KITTI 2," 2020.

[155] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 1–16, 2017.

[156] M. Wrenninge and J. Unger, "Synscapes: A Photorealistic Synthetic Dataset for Street Scene Parsing," 2018.

[157] A. R. Sekkat, Y. Dupuis, V. R. Kumar, H. Rashed, S. Yogamani, P. Vasseur, and P. Honeine, "SynWoodScape: Synthetic Surround-View Fisheye Camera Dataset for Autonomous Driving," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8502–8509, 2022.

[158] S. Ramachandran, N. Cibik, G. Sistu, and J. McDonald, "WoodScape Motion Segmentation for Autonomous Driving – CVPR 2023 OmniCV Workshop Challenge," 2024.

[159] C. Wu, "Visual SFM." http://ccwu.me/vsfm/doc.html.

[160] J. P. e. a. Cruise Automation, Audrey Li, "Webviz: Visualizing robotics data in the browser." https://webviz.io/.

[161] U. ATG and VIS.GL, "The Autonomous Visualization System." https://avs.auto.

[162] B. E. Moore and J. J. Corso, "FiftyOne," *GitHub. Note: https://github.com/voxel51/fiftyone*, 2020.

[163] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.

[164] P. Lee Clement, M. Nutt, jkelly stars, A. Aggarwal, N. Johnson, and A. Carballo, "utiasSTARS/pykitti." https://github.com/utiasSTARS/pykitti, 2023.

[165] M. Gadd, G. Pascoe, D. Barnes, S. Brahmbhatt, C. Ros, and Farid, "Robotcar Dataset SDK." https://github.com/ori-mrg/robotcar-dataset-sdk.

[166] M. Cordts, N. Gählert, jmtatsch, N. Jourdan, A. Kirillov, B. Luthra, L. Beyer, A. Rana, A. Köring, L. FAN, M. Teichmann, S. Rühle, ruic, and K. Rosaen, "mcordts/cityscapesScripts." https://github.com/mcordts/cityscapesScripts, 2024.

[167] F. Yu, X. Li, X. Wang, T. Fisher, and J. Pang, "BDD100K Toolkit." https://github.com/bdd100k/bdd100k.

[168] V. Gudivada, A. Apon, and J. Ding, "Data Quality Considerations for Big Data and Machine Learning: Going Beyond Data Cleaning and Transformations," *International Journal on Advances in Software*, vol. 10, pp. 1–20, 07 2017.

[169] R. Franklin, "How Data Quality impacts Machine Learning." https://www.precisely.com/blog/data-quality/data-quality-impact-machine-learning.

[170] G. Krasadakis, "Data Quality in the era of A.I.." https://www.freecodecamp.org/news/data-quality-in-the-era-of-a-i-d8e398a91bef/.

[171] T. C. Redman, "If your data is bad, your Machine Learning tools are useless." https://hbr.org/2018/04/if-your-data-is-bad-your-machine-learning-tools-are-useless.

[172] B. J. et al., "glMatrix, A Javascript Matrix and Vector library for High Performance WebGL apps." https://github.com/toji/gl-matrix.

[173] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, pp. 357–362, Sept. 2020.

[174] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[175] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[176] G. Guennebaud, B. Jacob, *et al.*, "Eigen v3." http://eigen.tuxfamily.org, 2010.

[177] M. Albéri, M. Baldoncini, C. Bottardi, E. Chiarelli, G. Fiorentini, *et al.*, "Accuracy of Flight Altitude Measured with Low-Cost GNSS, Radar and Barometer Sensors: Implications for Airborne Radiometric Surveys," *Sensors*, vol. 17, p. 1889, Aug. 2017.

[178] Garmin Ltd, "Understanding the Accuracy of the GPS Elevation Reading," 2005.

[179] Stéfan van der Walt and Nathaniel Smith, "A Better Default Colormap for Matplotlib." https://bids.github.io/colormap/, 2015.

[180] V. Agafonkin, "Leaflet - a JavaScript library for interactive maps." https://leafletjs.com.

[181] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ." https://www.openstreetmap.org, 2017.

[182] R. W. Sinnott, "Virtues of the Haversine," *skytel*, vol. 68, p. 158, Dec. 1984.

[183] D. Powers, "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

[184] J. Graver, "From the accidental to the deliberate." http://mortalmuses.com/2013/01/15/accidental-to-deliberate/, 2013.

[185] R. Dubé, A. Cramariuc, D. Dugas, J. I. Nieto, R. Siegwart, and C. Cadena, "SegMap: 3D Segment Mapping using Data-Driven Descriptors," *ArXiv*, vol. abs/1804.09557, 2018.

[186] S. Ramachandran and J. McDonald, "Place Recognition in Challenging Conditions," in *21st Irish Machine Vision and Image Processing Conference*, 8 2019.

[187] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[188] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *International conference on machine learning*, pp. 647–655, PMLR, 2014.

[189] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition," in *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 512–519, 2014.

[190] Z. Chen, O. Lam, A. Jacobson, and M. Milford, "Convolutional neural network-based place recognition," in *Proceedings of the 16th Australasian Conference on Robotics and Automation 2014* (C. Chen, ed.), pp. 1–8, Australia: Australian Robotics and Automation Association (ARAA), 2014.

[191] Y. Hou, H. Zhang, and S. Zhou, "Convolutional neural network-based image representation for visual loop closure detection," in *2015 IEEE international conference on information and automation*, pp. 2238–2245, IEEE, 2015.

[192] N. Sünderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford, "On the performance of convnet features for place recognition," in *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 4297–4304, IEEE, 2015.

[193] X. Zhang, Y. Su, and X. Zhu, "Loop closure detection for visual SLAM systems using convolutional neural network," in *2017 23rd International Conference on Automation and Computing (ICAC)*, pp. 1–6, IEEE, 2017.

[194] PapersWithCode, "Semantic Segmentation Benchmarks on PASCAL VOC 2012." https://paperswithcode.com/sota/semantic-segmentation-on-pascal-voc-2012, 2019.

[195] G. Chechik, V. Sharma, U. Shalit, and S. Bengio, "Large Scale Online Learning of Image Similarity Through Ranking," *J. Mach. Learn. Res.*, vol. 11, p. 1109–1135, Mar. 2010.

[196] F. Chollet *et al.*, "Keras." https://keras.io/preprocessing/image/#imagedatagenerator-class, 2015.

[197] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *2016 IEEE CVPR*, pp. 2818–2826, June 2016.

[198] S. H. Khan, M. Hayat, M. Bennamoun, R. Togneri, and F. A. Sohel, "A discriminative representation of convolutional features for indoor scene recognition," *IEEE TIP*, vol. 25, no. 7, pp. 3372–3383, 2016.

[199] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," *ArXiv*, vol. abs/1311.2901, 2013.

[200] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 618–626, 2017.

[201] B. Zhou, P. Krähenbühl, and V. Koltun, "Does computer vision matter for action?," *Science Robotics*, vol. 4, no. 30, p. eaaw6661, 2019.

[202] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[203] I. Biederman, "Aspects and extensions of a theory of human image understanding," *Computational processes in human vision: An interdisciplinary perspective*, pp. 370–428, 1988.

[204] C. Zhuang, A. L. Zhai, and D. Yamins, "Local aggregation for unsupervised learning of visual embeddings," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6002–6012, 2019.

[205] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," 2016.

[206] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning* (D. Precup and Y. W. Teh, eds.), vol. 70 of *Proceedings of Machine Learning Research*, pp. 214–223, PMLR, 06–11 Aug 2017.

[207] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation," 2018.

[208] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," 2021.

[209] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, (Red Hook, NY, USA), Curran Associates Inc., 2020.

[210] S. Ramachandran, J. Horgan, G. Sistu, and J. McDonald, "Fast and Efficient Scene Categorization for Autonomous Driving using VAEs," in *24th Irish Machine Vision and Image Processing Conference*, IMVIP 2022, Irish Pattern Recognition and Classification Society, Aug. 2022.

[211] L. Xie, F. Lee, L. Liu, K. Kotani, and Q. Chen, "Scene recognition: A comprehensive survey," *Pattern Recognition*, vol. 102, p. 107205, 2020.

[212] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3730–3738, 2015.

[213] S. Zhao, J. Song, and S. Ermon, "InfoVAE: Information Maximizing Variational Autoencoders," *CoRR*, vol. abs/1706.02262, 2017.

[214] A. Kumar, P. Sattigeri, and A. Balakrishnan, "Variational inference of disentangled latent concepts from unlabeled observations," *arXiv preprint arXiv:1711.00848*, 2017.

[215] A. Amini *et al.*, "Variational autoencoder for end-to-end control of autonomous driving with novelty detection and training de-biasing," in *2018 IEEE/RSJ IROS*, pp. 568–575, IEEE, 2018.

[216] M. Uricar, G. Sistu, H. Rashed, A. Vobecky, V. R. Kumar, P. Krizek, F. Burger, and S. Yogamani, "Let's get dirty: GAN based data augmentation for camera lens soiling detection in autonomous driving," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 766–775, 2021.

[217] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 413–420, IEEE, 2009.

[218] V. Lempitsky, A. Vedaldi, and D. Ulyanov, "Deep Image Prior," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9446–9454, 2018.

[219] A. L. Maas, A. Y. Hannun, A. Y. Ng, *et al.*, "Rectifier nonlinearities improve neural network acoustic models," in *ICML*, vol. 30, p. 3, Citeseer, 2013.

[220] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, pp. 448–456, PMLR, 2015.

[221] I. Higgins *et al.*, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *ICLR 2017*, 2016.

[222] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[223] X. Hou, L. Shen, K. Sun, and G. Qiu, "Deep feature consistent variational autoencoder," in *2017 IEEE WACV*, pp. 1133–1141, IEEE, 2017.

[224] P. Chen, G. Chen, and S. Zhang, "Log Hyperbolic Cosine Loss Improves Variational Auto-Encoder," 2019.

[225] T. Rainforth, A. Kosiorek, T. A. Le, C. Maddison, M. Igl, F. Wood, and Y. W. Teh, "Tighter variational bounds are not necessarily better," in *ICML*, pp. 4277–4285, PMLR, 2018.

[226] H. Korrapati and Y. Mezouar, "Vision-based sparse topological mapping," *Robotics and Autonomous Systems*, vol. 62, no. 9, pp. 1259–1270, 2014.

[227] S. Ramachandran, J. Horgan, G. Sistu, and J. McDonald, "Scalable and Efficient Hierarchical Visual Topological Mapping," in *2023 21st International Conference on Advanced Robotics (ICAR)*, pp. 113–120, 2023.

[228] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof, "From structure-from-motion point clouds to fast location recognition," in *2009 IEEE CVPR*, pp. 2599–2606, IEEE, 2009.

[229] S. Middelberg, T. Sattler, O. Untzelmann, and L. Kobbelt, "Scalable 6-dof localization on mobile devices," in *European conference on computer vision*, pp. 268–283, Springer, 2014.

[230] P.-E. Sarlin, F. Debraine, M. Dymczyk, R. Siegwart, and C. Cadena, "Leveraging deep visual descriptors for hierarchical efficient localization," in *Conference on Robot Learning*, pp. 456–465, PMLR, 2018.

[231] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *IEEE CVPR*, pp. 779–788, 2016.

[232] S. Ramachandran, G. Sistu, J. McDonald, and S. Yogamani, "Woodscape Fisheye Semantic Segmentation for Autonomous Driving – CVPR 2021 OmniCV Workshop Challenge," 2021.

[233] S. Ramachandran, G. Sistu, V. R. Kumar, J. McDonald, and S. Yogamani, "Woodscape Fisheye Object Detection for Autonomous Driving – CVPR 2022 OmniCV Workshop Challenge," 2022.

[234] C. Toft, E. Stenborg, L. Hammarstrand, L. Brynte, M. Pollefeys, T. Sattler, and F. Kahl, "Semantic match consistency for long-term visual localization," in *Proceedings of ECCV*, pp. 383–399, 2018.

[235] J. Bai, F. Lu, K. Zhang, *et al.*, "ONNX: Open Neural Network Exchange." https://github.com/onnx/onnx, 2019.

[236] R. Arroyo, P. F. Alcantarilla, L. M. Bergasa, J. J. Yebes, and S. Bronte, "Fast and effective visual place recognition using binary codes and disparity information," in *IEEE/RSJ IROS*, pp. 3089–3094, 2014.

[237] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning Transferable Visual Models From Natural Language Supervision," in *Proceedings of the 38th International Conference on Machine Learning* (M. Meila and T. Zhang, eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763, PMLR, 18–24 Jul 2021.

[238] J. Li, D. Li, C. Xiong, and S. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation," in *International conference on machine learning*, pp. 12888–12900, PMLR, 2022.

[239] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," in *International conference on machine learning*, pp. 19730–19742, PMLR, 2023.

[240] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked Autoencoders Are Scalable Vision Learners," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15979–15988, 2022.

[241] Y. Li, G. Yuan, Y. Wen, J. Hu, G. Evangelidis, S. Tulyakov, Y. Wang, and J. Ren, "Efficientformer: Vision transformers at mobilenet speed," *Advances in Neural Information Processing Systems*, vol. 35, pp. 12934–12949, 2022.

[242] Y. Tay, M. Dehghani, S. Abnar, Y. Shen, D. Bahri, P. Pham, J. Rao, L. Yang, S. Ruder, and D. Metzler, "Long Range Arena : A Benchmark for Efficient Transformers," in *ICLR 2021*, 2021.

[243] C. Zhu, W. Ping, C. Xiao, M. Shoeybi, T. Goldstein, A. Anandkumar, and B. Catanzaro, "Long-short transformer: Efficient transformers for language and vision," *Advances in neural information processing systems*, vol. 34, pp. 17723–17736, 2021.

[244] N. Kitaev, L. Kaiser, and A. Levskaya, "Reformer: The Efficient Transformer," *ArXiv*, vol. abs/2001.04451, 2020.

[245] S. Hu, H. Fan, B. Gao, X. Zhao, and H. Zhao, "An Image-based Approach of Task-driven Driving Scene Categorization," *CoRR*, vol. abs/2103.05920, 2021.

[246] I. Sikirić, K. Brkić, P. Bevandić, I. Krešo, J. Krapac, and S. Šegvić, "Traffic Scene Classification on a Representation Budget," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 1, pp. 336–345, 2020.

[247] S. Varghese, S. Gujamagadi, M. Klingner, N. Kapoor, A. Bar, J. D. Schneider, K. Maag, P. Schlicht, F. Huger, and T. Fingscheidt, "An Unsupervised Temporal Consistency (TC) Loss To Improve the Performance of Semantic Segmentation Networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 12–20, June 2021.

[248] Y. Zhu, K. Sapra, F. A. Reda, K. J. Shih, S. Newsam, A. Tao, and B. Catanzaro, "Improving Semantic Segmentation via Video Propagation and Label Relaxation," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8848–8857, 2019.

[249] P.-Y. Chen, A. H. Liu, Y.-C. Liu, and Y.-C. F. Wang, "Towards Scene Understanding: Unsupervised Monocular Depth Estimation With Semantic-Aware Representation," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2619–2627, 2019.

[250] G. Csurka, *Domain Adaptation in Computer Vision Applications*. Springer International Publishing, 2017.

[251] A. Farahani, S. Voghoei, K. Rasheed, and H. R. Arabnia, "A brief review of domain adaptation," in *Advances in Data Science and Information Engineering* (R. Stahlbock, G. M. Weiss, M. Abou-Nasr, C.-Y. Yang, H. R. Arabnia, and L. Deligiannidis, eds.), (Cham), pp. 877–894, Springer International Publishing, 2021.

[252] M. Toldo, A. Maracani, U. Michieli, and P. Zanuttigh, "Unsupervised Domain Adaptation in Semantic Segmentation: A Review," *Technologies*, vol. 8, no. 2, 2020.

[253] M. Galarnyk, N. Cibik, O. Maher, and P. Thomas, "Synthetic Data Best Practices for Perception Applications." Parallel Domain Blog, 2023.

[254] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superglue: Learning feature matching with graph neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4938–4947, 2020.

[255] B. Cao, A. Araujo, and J. Sim, "Unifying deep local and global features for image search," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pp. 726–743, Springer, 2020.

[256] U. Efe, K. G. Ince, and A. Alatan, "DFM: A performance baseline for deep feature matching," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4284–4293, 2021.

[257] Y. Ono, E. Trulls, P. Fua, and K. M. Yi, "LF-Net: Learning local features from images," *Advances in neural information processing systems*, vol. 31, 2018.

[258] L. Deng, G. Li, S. Han, L. Shi, and Y. Xie, "Model Compression and Hardware Acceleration for Neural Networks: A Comprehensive Survey," *Proceedings of the IEEE*, vol. 108, no. 4, pp. 485–532, 2020.

[259] G. Fang, X. Ma, M. Song, M. B. Mi, and X. Wang, "Depgraph: Towards any structural pruning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16091–16101, 2023.

[260] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," *arXiv preprint arXiv:1611.06440*, 2016.

[261] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2704–2713, 2018.

[262] Z. Liu, Y. Wang, K. Han, W. Zhang, S. Ma, and W. Gao, "Post-training quantization for vision transformer," *Advances in Neural Information Processing Systems*, vol. 34, pp. 28092–28103, 2021.

[263] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4820–4828, 2016.

[264] R. Krishnamoorthi, "Quantizing deep convolutional networks for efficient inference: A whitepaper," *arXiv preprint arXiv:1806.08342*, 2018.

[265] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. Van Baalen, and T. Blankevoort, "A white paper on neural network quantization," *arXiv preprint arXiv:2106.08295*, 2021.

[266] T. Lin, S. U. Stich, L. Barba, D. Dmitriev, and M. Jaggi, "Dynamic model pruning with feedback," *arXiv preprint arXiv:2006.07253*, 2020.

[267] S. Laskaridis, S. I. Venieris, H. Kim, and N. D. Lane, "HAPI: Hardware-Aware Progressive Inference," in *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pp. 1–9, 2020.

[268] S. Laskaridis, A. Kouris, and N. D. Lane, "Adaptive inference through early-exit networks: Design, challenges and directions," in *Proceedings of the 5th International Workshop on Embedded and Mobile Deep Learning*, pp. 1–6, 2021.

[269] T. Bolukbasi, J. Wang, O. Dekel, and V. Saligrama, "Adaptive neural networks for efficient inference," in *International Conference on Machine Learning*, pp. 527–536, PMLR, 2017.

[270] N. Passalis, J. Raitoharju, A. Tefas, and M. Gabbouj, "Efficient adaptive inference for deep convolutional neural networks using hierarchical early exits," *Pattern Recognition*, vol. 105, p. 107346, 2020.

[271] E. Bengio, P.-L. Bacon, J. Pineau, and D. Precup, "Conditional computation in neural networks for faster models," *arXiv preprint arXiv:1511.06297*, 2015.

[272] A. Veit and S. Belongie, "Convolutional networks with adaptive inference graphs," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–18, 2018.

[273] M. Figurnov, M. D. Collins, Y. Zhu, L. Zhang, J. Huang, D. Vetrov, and R. Salakhutdinov, "Spatially adaptive computation time for residual networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1039–1048, 2017.