APPEARANCE SLAM IN CHANGING ILLUMINATION ENVIRONMENT

by

Yang Liu

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science
University of Alberta

Abstract

With the rapid development in visual sensors such as monocular vision, appearance-based robot simultaneous localization and mapping (SLAM) has become an open research topic in robotics. In appearance SLAM, a robot uses the visual appearance of locations (i.e., the images) acquired along its route to build a map of the environment and localizes itself by recognizing the places it has visited before. In this thesis, we address several issues in the current appearance SLAM techniques, with the intention to develop a systematic approach for SLAM under significant illumination change – a typical scenario in long-term mapping. Instead of using traditional Bag-of-Words (BoW) image descriptor in comparing the appearance of locations, we use visual features directly to solve the perceptual aliasing that may particularly happen in illumination change caused partially by vector quantization of feature descriptors in image encoding. Efficient data structures such as *k*-d tree or random *k*-d forests are exploited to speed up the feature matching with approximate nearest neighbor search to ensure real-time robot exploration, without sacrificing performance at the level of matching locations.

In order to deal with the cases in which local features do not work well, for example, in the environment with significant illumination variance where feature repeatability is not guaranteed, we propose to use a whole-image descriptor which is a low dimensional compact representation of image responses to a bank of filters incorporating the structural information (e.g. the edges) of an image to describe the appearance and measure similarities among locations. PCA is employed to transform a high dimensional gist descriptor to a lower dimensional form to improve both computational efficiency and discriminating power of the descriptor. In addition, we use a particle filter to exploit the correlation among images in a sequence captured by the robot in the process of identifying loop closure candidates, making the algorithm highly scalable due to both the compactness of image descriptor and simplicity of particle filtering.

Based on the above methods, our final component of the SLAM system is a novel feature matching method for multi-view geometry (MVG) based verification of loop closures in illumination change. To develop such a method that serves as the prerequisite of verification, we exploit the particular camera motion in our application to illustrate that spatial constraint of matching features (or keypoints) derived from optical flow statistics can be used as an important basis in finding true matches. Particularly, by assuming a weak perspective camera model and planar camera motion, we derive a simple constraint on correctly matched keypoints in terms of the flow vectors between two images. We then use this constraint to prune the putative matches to boost the inlier ratio significantly thereby giving the subsequent verification algorithm a chance to succeed.

$\sim$

Preface

Most of the research described in this thesis has been previously published. Chapter 2 of this thesis has been published as Y. Liu and H. Zhang, “Indexing Visual Features: Real-Time Loop Closure Detection Using a Tree Structure,” *IEEE International Conference on Robotics and Automation*, May 2012. I was responsible for the implementation, analysis and manuscript composition. H. Zhang assisted with the algorithm design and contributed to manuscript edits. Chapter 3 of this thesis has been published as Y. Liu and H. Zhang, “Visual Loop Closure Detection with a Compact Image Descriptor,” *IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2012, and Y. Liu and H. Zhang, “Towards Improving the Efficiency of Sequence-Based SLAM,” *IEEE International Conference on Mechatronics and Automation*, August 2013. I was responsible for the implementation, analysis and manuscript composition. H. Zhang assisted with the algorithm design and contributed to manuscript edits. Chapter 4 of this thesis has been published as Y. Liu and H. Zhang, “Performance Evaluation of Whole-Image Descriptors in Visual Loop Closure Detection,” *IEEE International Conference on Information and Automation*, August 2013. I was responsible for the implementation and manuscript composition. H. Zhang contributed to manuscript edits. Chapter 5 of this thesis has been published as Y. Liu, R. Feng and H. Zhang, “Keypoint Matching by Outlier Pruning with Consensus Constraint,” *IEEE International Conference on Robotics and Automation*, May 2015. I was responsible for the data collection, algorithm design, implementation and manuscript composition. R. Feng contributed partially to the experiment. H. Zhang assisted with the algorithm design and contributed to manuscript edits.

$\sim$

Acknowledgement

Before starting a PhD journey, people always think that they are so clear about what they are doing and therefore extremely ambitious and optimistic towards achieving the goals they have set. As time goes by then, many begin to realize that their confidence, spirit and enthusiasm gradually wear out before totally getting lost. Fortunately, I do not belong to one of those, simply because of my PhD supervisor, Professor Hong Zhang, who has led me to the world of research and shown me the direct and clear path to follow during the entire process. I must express my sincere appreciation to Hong, not only for the reason of being my supervisor, but also due to the fact that he has granted me a particular set of skills, methodologies, and most importantly, attitudes when facing difficulties and solving problems in life. For the past five years under the guidance of Hong, I have remoulded myself both mentally and technically, being well-prepared for the near future career in a competitive environment. I think that is the best a PhD student can expect.

I give my great thanks to my committee, including Dr. Nilanjan Ray, Dr. Martin Jagersand, Dr. Martin Mueller and Dr. Ryan Eustice. Their hard work and feedback have significantly facilitated the completion of my thesis. Every single question has assisted me identifying the weakness and shortcoming not only in the thesis, but also of myself.

I have spent a wonderful and unforgettable five years in CIMS and Robotics labs, together with my current colleagues and the alumnus who are continuously making contributions to the society with their talents and devotions. Without the happiness and help they have brought to the labs, work could have become tedious and inefficient. While there is no never-ending feast, the good old days accompanied with their smiles will eternally stay in my mind and become my best memories. So thank you all.

Finally, my utmost gratitude goes to my family – my parents in China and my wife Qing and son Jason. Their unconditional support and love have made my every step in pursuing the degree possible. When I was spending money immoderately, my father was working day and night in the hospital, conducting surgeries one after another and taking care of the patients without one minute break, even in New Year's Eve, to earn money for me to spend. When I was travelling for fun insouciantly, my mother was thinking about my health and safety every moment, expecting my return. For the past thirty years of my life, I have never provided them anything substantial, not a single penny. I have also owed my wife and son too much. While the other families are enjoying the festivals, the very basic expectation of staying with the husband and dad becomes a luxury for them. As my gratitude cannot be described in literacy, I promise to be a qualified son, husband and father from this moment on.

$\sim$

Table of Contents

List of Tables

List of Figures

List of Symbols

Notation	Definition
$\mathbf{x}_k$	Robot pose, including location and orientation
$\mathbf{u}_k$	Control input, applied at time $k-1$ to drive the robot to a state $\mathbf{x}_k$
$\mathbf{m}_i$	location of the i^{th} landmark
$\mathbf{z}_k$	An observation taken from the robot at time k
$\mathbf{X}_{0:k} = \{\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_k\}$	The history of robot locations
$\mathbf{U}_{0:k} = \{\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_k\}$	The history of control inputs
$\mathbf{m} = \{\mathbf{m}_0, \mathbf{m}_1, \cdots, \mathbf{m}_n\}$	The set of all landmarks
$\mathbf{Z}_{0:k} = \{\mathbf{z}_0, \mathbf{z}_1, \cdots, \mathbf{z}_k\}$	The set of all landmark observations

Chapter 1

Introduction

1.1 Foreword

A mobile robot can perform a wide variety of tasks including exploration, detection, operation, etc., in complex or dangerous environments where human beings cannot be involved directly. As the research of robotics develops rapidly, mobile robots are now playing an important role in traditional industries such as manufacturing, logistics, services, etc. There have been several instances where a mobile robot can be used to replace human beings towards improving the efficiency and productivity. In addition, the needs for mobile robots increase significantly in a particular set of programs that are strategically crucial to economics, social security and national defense. With the continuous expansion of human activities, mobile robots are making considerable influences and showing vast potentialities in areas such as planet exploration, military reconnaissance, anti-terrorism, disaster rescue, ocean exploitation, hazard handling, etc. Many of these applications are in unknown dynamic environments incorporating variances such as in illumination, scene and object motion. Therefore, the research of mobile robotics in these environments is important and meaningful. In this thesis, we focus on the research of vision-based robot navigation in case of illumination change, as described later.

1.2 Research Background

In the past few decades, navigation has become one of the most important research topics in the robotics community. The ability of accurate navigation is considered a prerequisite for a mobile robot to be truly autonomous in an unknown environment where the robot location and map information cannot be obtained from external sensors such as Global Positioning System (GPS). In order to achieve successful navigation, a robot has to perform simultaneous localization and mapping (SLAM) to both identify its location in the environment and incrementally build the map at the same time.

1.2.1 SLAM at a Glance

A milestone in SLAM is the research of Smith and Cheeseman on the representation and estimation of spatial uncertainty [104] in 1986, in which a statistical basis was proposed for describing relationships between landmarks and manipulating geometric uncertainty. This soon led to the seminal work proposed in 1990 by Smith, Self and Cheeseman in their landmark paper [105], where stochastic mapping was achieved based on extended Kalman filter (EKF). In SLAM, a robot starts from an unknown location and detects landmarks in the environment with its sensors. Localization and mapping are performed online using the sensor measurements of the relative positions between the robot and landmarks. Therefore, the two processes are correlated and achieved at the same time. In general the following steps are iterated: 1) the robot predicts its pose and landmark positions for the next time step based on the current pose and map information, 2) it makes observations of landmarks using its sensors and 3) upon receiving the measurements from sensors, it corrects the predicted results of its pose and landmark positions. The estimation of robot pose and landmark positions gradually becomes accurate as the iteration proceeds, and mapping is completed incrementally. As all quantities in the estimation are usually probabilistic [111–113], the SLAM process can be modeled as a state estimation problem where the joint posterior of the system state $P(\mathbf{x}_k, \mathbf{m} \mid \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0)$ is estimated. Starting with an estimation $P(\mathbf{x}_{k-1}, \mathbf{m} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}, \mathbf{x}_0)$ at $k-1$, the posterior is calculated in two steps

Prediction

$$\begin{aligned} &P(\mathbf{x}_k, \mathbf{m} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0) \\ &= \int P(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) P(\mathbf{x}_{k-1}, \mathbf{m} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}, \mathbf{x}_0)\, \mathrm{d}\mathbf{x}_{k-1} \end{aligned} \tag{1.1}$$

Update

$$P(\mathbf{x}_k, \mathbf{m} \mid \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) = \frac{P(\mathbf{z}_k \mid \mathbf{x}_k, \mathbf{m}) P(\mathbf{x}_k, \mathbf{m} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0)}{P(\mathbf{z}_k \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})} \tag{1.2}$$

Equation (1.1) and (1.2) are based on Bayes theorem and provide a recursive procedure for calculating $P(\mathbf{x}_k, \mathbf{m} \mid \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0)$. The recursion is a function of a motion model $P(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k)$ representing robot kinematics, and an observation model $P(\mathbf{z}_k \mid \mathbf{x}_k, \mathbf{m})$ [4, 32]. In order to solve this problem, an appropriate representation of $P(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k)$ and $P(\mathbf{z}_k \mid \mathbf{x}_k, \mathbf{m})$ is required. The most common one is in the form of a state space model with Gaussian noise, leading to the use of EKF which is a parametric method with closed-form solution to state estimation. By assuming that the joint system state consisting of both $\mathbf{x}_k$ and $\mathbf{m}$ follows a Gaussian distribution, both observation and motion models can be linearized using first order Taylor series expansion (note that a robot system is non-linear due to its observation model and motion in orientation) and system state can be estimated using standard Kalman filter (KF) procedures. Therefore, the core issue in EKF is to calculate the weighted average values (the numerator in Equation (1.2)) of the system state and the associated covariance matrix representing the system uncertainty caused by the assumed Gaussian noise.

Alternatively, FastSLAM [77–79] proposed by Montemerlo *et al.* based on Rao-Blackwellised particle filter [31, 85] built another milestone in terms of implementing the recursive Bayes filtering in the context of probabilistic SLAM. The direct use of Monte Carlo sampling [30, 63], or particularly particle filter [34] to SLAM is not feasible due to the super high dimensional state space whose probabilistic distribution cannot be approximated by sampling with a finite number of particles. However, it is possible to reduce the sample space by applying Rao-Blackwellisation to the joint state so that only a low dimensional space needs to be sampled. In SLAM, the joint state can be factored into a robot component and a conditional map component and a key observation is that when conditioned on the robot trajectory, the map landmarks become independent. Consequently, a particle filter can be used to estimate the robot trajectories, each of which is associated with a set of landmarks whose positions can be estimated with EKF. The joint distribution is represented as a set of weighted trajectories incorporating robot poses, together with their associated landmark estimations consisting of independent Gaussian distributions. FastSLAM was the first to directly represent the non-linear process model and non-Gaussian pose distribution. The biggest advantage however, is the efficiency in time complexity as the map estimation becomes linear (or even logarithmic) after factorization, rather than quadratic in the traditional EKF due to the joint map covariance.

The approaches mentioned above dominated the SLAM research in the early years. Many KF or EKF variants were developed to address different issues in the original algorithm. Well-known examples include unscented Kalman filter (UKF) [48,49], sparse extended information filter (SEIF) [115], exactly sparse extended information filter (ESEIF) [123], invariant extended kalman filter (IEKF) [9], observability constrained (OC)-EKF [45], compressed EKF [38,39], etc.. In general, EKF-based methods assume that geometric landmarks (or features) can be extracted from the environment and the map is defined as a dense set of such salient landmarks that may consist of points, edges or line segments, corners and planes [10, 95]. Map is usually maintained by recording the metrical information acquired by range-bearing sensors such as laser range finder or sonar (visual sensors such as monocular vision could be also used as in [27]). This kind of map representation is compact and can facilitate the estimation within the filtering framework. However, the features themselves can be difficult to extract, and the sensor information may need further processing. In addition, the number of landmarks may increase considerably as the size of map grows, making the algorithm intractable in large scale environment due to the computational burden.

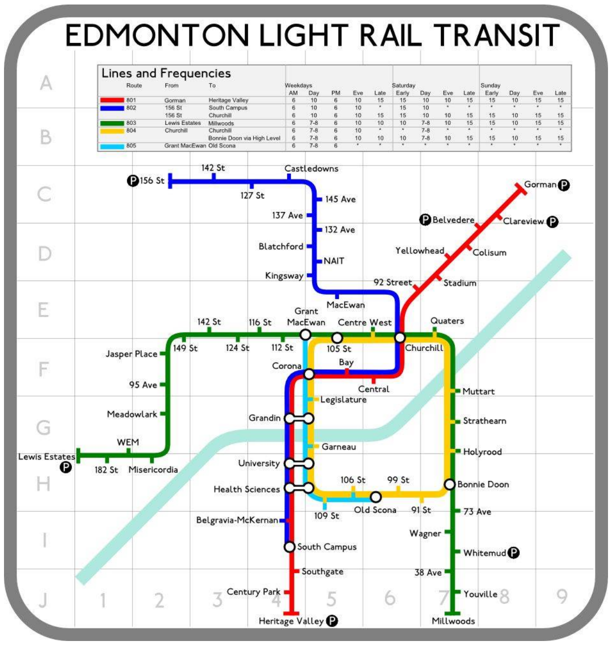
Thanks to the rapid development in visual sensors such as monocular vision, the trend of SLAM research in the recent years has turned to representing the environment using its appearance without being dependent on geometric landmarks. This type of representation can make a SLAM algorithm highly scalable in large maps. The use of Bayes filtering described in Equation (1.1) and (1.2) still applies to the estimation of robot location. However, as the landmarks are no longer defined during localization and mapping, the system uncertainty reflected by the covariance matrix in the traditional EKF-based methods is not explicitly maintained. As a result, the system update does not require a quadratic process of recalculating the uncertainty. Details of appearance SLAM are discussed in the following section.

1.2.2 SLAM in Appearance Space

Appearance SLAM is one type of visual SLAM using visual sensors. Different from many visual SLAM methods that still detect landmarks with depth (range) information obtained by stereo vision, SLAM using appearance builds a map with images. In general, the robot moves along its route and continuously takes images as the appearance of locations. Mapping is achieved by connecting these locations according to how similar they look, i.e., two images with sufficient similarity reflecting the same location are related to each other. Meanwhile, the robot localizes itself within this map by recognizing places it has visited before. Localization is essentially a place recognition problem, which is equivalent to loop closure detection in the context of SLAM. As a result, the map built in appearance SLAM is actually a topological graph of the places being the nodes of the graph. Only localization in the level of identifying locations is required, metrical information of the map such as the specific distance between two locations is optional, or usually ignored, as one example shown in Figure 1.1, the map of Edmonton Light Rail Transit (LRT) system. Under such a map representation, the core issue in appearance SLAM is to match locations, i.e., the images. Image matching usually requires two steps of 1) encoding the image in a particular form and 2) comparing images with their encoded representation.

Image Encoding

As a digital image in its original form usually cannot be directly used in matching due to both its high dimensionality and redundant information it may convey, an encoding scheme of transforming it to a more efficient and effective representation is desired. A common solution is to detect features in the image that can be characterized using feature descriptors. Early approaches focused on detecting edges [13] consisting of a set of pixels which have a strong gradient magnitude. However,



Lines and Frequencies

	Route	From	To	Weekdays AM	Day	PM	Eve	Late	Saturday Early	Day	Eve	Late	Sunday Early	Day	Eve	Late
🟥	801	Gorman	Heritage Valley	6	10	6	10	15	15	10	10	15	15	10	15	15
🟦	802	156 St	South Campus	6	10	6	10	*	15	10	*	*	*	*	*	*
		156 St	Churchill	6	10	6	10	15	15	10	10	15	15	10	15	15
🟩	803	Lewis Estates	Millwoods	6	7-8	6	10	10	10	7-8	10	15	15	10	15	15
🟨	804	Churchill	Churchill	6	7-8	6	10	*	*	7-8	*	*	*	*	*	*
			Bonnie Doon via High Level	6	7-8	6	10	10	10	7-8	10	15	15	10	15	15
🟦	805	Grant MacEwan	Old Scona	6	7-8	6	*	*	*	*	*	*	*	*	*	*

Figure 1.1: An example of a topological map of Edmonton LRT system (conceptual). In this map, no distance is given between locations. However, it does not affect us understanding the map.

edge features are not desirable in matching as it is difficult to properly describe a spatially continuous set of points. Alternatively, methods detecting blobs/regions of interest points are more popular. We briefly summarize the commonly used feature detectors below.

- **Scale-invariant region detectors**. This type of feature detectors generally involve two steps. First, an image is convolved by a Gaussian kernel at a certain scale to generate the scale-space representation of the original image. Then an operator is applied to the convolved image to obtain a multi-scale blob detector to detect scale-space extrema, resulting in strong responses for salient blobs. One of the first common operators is the Laplacian operator with automatic scale selection proposed by Lindeberg [58, 60], and the detector is named Laplacian of Gaussian (LoG). Alternatively, difference of Gaussians (DoG) is a simplified version of LoG and for instance used in the well-known scale-invariant feature transform (SIFT) algorithm proposed by Lowe [64, 65]. The change is that the Laplacian operator in LoG is computed as the limit case of the difference between two Gaussian smoothed images.

Another detector is determinant of Hessian (DoH) in which Laplacian operator is replaced with the scale-normalized determinant of the Hessian matrix of the scale-space representation. An integer approximation of DoH computed from Haar wavelets is used as the basic interest point operator in speeded up robust features (SURF) [6]. It is shown in [59] that DoH performs significantly better than LoG and DoG for image matching using local SIFT-like image descriptors. In addition, spatial selection can be done by DoH, and scale selection with scale-normalized Laplacian [69], resulting in a hybrid operator between the Laplacian and DoH named Hessian-Laplace.

- **Affine-invariant region detectors**. The above methods can generate features that are invariant to translation and scale changes. It is more important to find features that can be extracted under large viewpoint change related by affine transformations. To this end, many feature detectors have been developed using affine shape adaption [5, 61, 69, 120] to incorporate affine transformations, with the examples of Harris and Hessian affine region detectors, both of which were proposed by Mikolajczyk and Schmid [68]. Some other examples of this type of detectors include maximally stable extremal regions (MSER) [67], Kadir-Brady saliency detector [51, 52], edge-based regions (EBR) [120], intensity-extrema-based regions (IBR) [106, 119] and principal curvature-based region detector (PCBR) [29]. A detailed comparison of most affine-invariant detectors is provided in [71]. We also mention the features from accelerated segment test (FAST) detector originally proposed by Rosten and Drummond [96]. It does not belong to any of the above categories but is computationally efficient due to simple pixel intensity comparison.

Once a set of interest regions are extracted, a subsequent step is to encode their content in feature descriptors for robust matching. This is usually done by defining a local area around a detected keypoint and describing this area with gradient or intensity. For example, SIFT [64, 65] computes a gradient orientation histogram for a 16×16 region around a keypoint, partitioned to 16 (4×4) subregions, each of which using 8 bins as the quantization of orientations. The dimensionality of the descriptor is therefore 128 ($4 \times 4 \times 8$). Similar descriptors using gradient are gradient location and orientation histogram [70], PCA-SIFT [54] and histogram of oriented gradients [25]. Alternatively, SURF [6] uses the sum of Haar wavelet response around the keypoint, which is essentially the gradient of a small neighborhood of pixels. More recently, binary descriptors such as binary robust independent elementary features (BRIEF) [12], binary robust invariant scalable keypoints (BRISK) [57] and oriented FAST and rotated BRIEF (ORB) [98] were proposed. These descriptors are generated by comparing pairwise pixel intensities within a random or predefined pattern. The main advantage is that they are extremely fast to compute and highly compact, with comparable performance to the real-valued descriptors. Performance evaluation of feature descriptors are provided in [70] and [44].

Feature detectors and descriptors together encode an image to a set of vectors with either real values or binary code. While this type of encoding works well for many applications due to the preserved local invariance in scale, rotation, translation, illumination, etc., the potential limitations include 1) time-consuming in feature detection and matching, 2) memory inefficient and 3) not robust to significant illumination change (e.g. between sunny day and dark night) as both feature repeatability and descriptor variance may become an issue. An alternative of encoding an image is to use the whole-image descriptor which captures the general structure of the scene in the image without focusing too much on local details. A whole-image descriptor can be computed using the image responses to a bank of filters (e.g. Gabor filters with different orientations and frequencies) that preserve the texture information [88]. It can also be obtained by transforming each pixel to a local binary pattern code [125], or directly using feature descriptors in the entire image, with the image center as a "virtual keypoint" [3, 109]. Color histogram in the early work [121] shows the possibility of applying a simple whole-image descriptor in robotics for place recognition. Even simpler, an image itself can be downsampled to a smaller size and used as a descriptor with values being either the normalized pixel intensities or their binarization [73, 74, 76, 126]. The whole-image descriptors mentioned here have all been successfully used in robotics applications, particularly, appearance SLAM or place recognition [84, 100, 101, 117].

Image Comparison

Image comparison is a subsequent step after encoding an image and generates the candidate(s) of being a loop closure (a visited place) or new location. If a whole-image descriptor is used, comparison is usually done by calculating the Euclidean distance between two descriptor vectors. On the other hand, comparison using local invariant features involves feature matching between two sets of keyponits. For a keypoint in one image, its nearest neighbor in the other image is deemed to be a match, conditioned on some uniqueness constraints [65]. As the current frame is compared with all the existing map nodes, this nearest neighbor search soon becomes intractable as map grows. To address this problem, approximate nearest neighbor search needs to be applied, which requires an efficient data structure such as *k*-d tree [80–82], randomized *k*-d forest [90, 102], locality-sensitive hashing [26] and *k*NN graph [40]. Some examples of appearance SLAM using feature matching can be found in [53, 128, 129] and there also have been many instances of using these data structures in visual search of keypoints [8, 11, 24, 36, 41, 83].

Bag-of-Words in Appearance SLAM

Based on the image encoding scheme introduced in Section 1.2.2, a significant improvement is to apply the idea of inverted index from text indexing [130] to image retrieval [103]. The image descriptors are assigned by vector quantization to corresponding visual words from a visual vocabulary (Bag-of-Words, BoW) generated offline by clustering a pool of features. Image retrieval then becomes an indexing procedure where only relevant database images containing the same visual words are returned. The method avoids large scale search in direct feature matching and reduces comparison time to constant, depending only on the predefined number of visual words. This soon leads to the novel solution to appearance SLAM using BoW, with representative examples proposed by Cummins *et al* [23, 24] and Angeli *et al* [1, 2]. As mentioned previously, a tree structure [87] can be used to organize the visual words to speed up the search if a large number of words are used [24]. Binary visual words are also possible to provide satisfactory performance with faster image comparison in appearance SLAM [35, 36]. A well-known variant of BoW is proposed in [55].

Place Recognition in Illumination Change

In appearance SLAM, image encoding and comparison form the fundamental steps. Loop closure detection, the important issue of identifying a previously visited location, then becomes a place recognition problem. Early work in place recognition [37, 121, 127] was not discussed in the context of SLAM. However, as the research of SLAM develops rapidly, place recognition has been combined with appearance SLAM and there is a continuous focus on the problem in illumination change in long-term autonomy. Current solutions can be categorized into the following types.

- **Direct feature matching**. This has been discussed in Section 1.2.2. The basis idea is to use original keypoint descriptors instead of their vector quantized representation in BoW. A performance evaluation of SIFT and SURF in seasonal change is provided in [122], and an example of feature matching in appearance SLAM in dynamic environment can be found in [53].

- **Learning the change**. This is the most commonly used approach. Learning can be done in local feature level, i.e., a visual vocabulary reflecting the co-occurrence rate of two visual words corresponding to the same landmark (but may be in different illumination conditions) is learned offline to characterize the relationship among all the visual words [72]. Image comparison in different illumination is then possible because similar images are related due to containing visual words of the same landmark, even if they are taken in various illumination [14,46,47,94]. Alternatively and similarly, learning can be also achieved in whole-image descirptor level [108]. This type of methods usually require a large amount of labelled training data for learning to be feasible [22,92].

- **Using abundant sensory information**. A direct way of obtaining the information is to use sensor fusion to combine different types of sensors as in [75]. Alternatively, an image sequence, rather than a single image, can be used in matching [76, 89] to boost the performance. The basic assumption is that when the current observation is too poor or inaccurate to use in significant illumination change, there must be useful information obtaining from other sources to compensate the defected observation. In fact, sequence-based matching has shown promising results in not only illumination change, but also seasonal change that involves life-long localization and mapping [107].

In a nutshell, appearance SLAM is currently a thriving area in vision-based mobile robot navigation. The challenging situation of illumination change in long-term autonomy has attracted an increasing amount of attention. Promising results have been shown. However, some open problems still exist and new research topics are continuously emerging. Based on this background, this thesis studies the related issues in appearance SLAM, especially in illumination changing environment, trying to obtain substantial results and provide impetus to the research and development of the field.

1.2.3 Potential Problems

Appearance SLAM has become a major topic in the research of vision-based robot navigation and a considerable amount of results have been achieved. In terms of research and application, the following limitations and problems are still present.

Perceptual Aliasing

Perceptual aliasing [23] is the most common and intrinsic problem existing in appearance SLAM, especially in environment with significant illumination variance or other types of dynamic change. It refers to the problem of high ambiguity between locations: images representing totally different places are considered correct matches, or conversely, a true matching location is considered a different place. Figure 1.2 shows two examples of perceptual aliasing, where (a) and (b) are separated places with similar appearance, and (c) and (d) are taken from the same spot but look differently due to illumination change. In a traditional BoW image encoding method, perceptual aliasing can be partially caused by vector quantization of local invariant feature descriptors. The visual words in a vocabulary actually form a

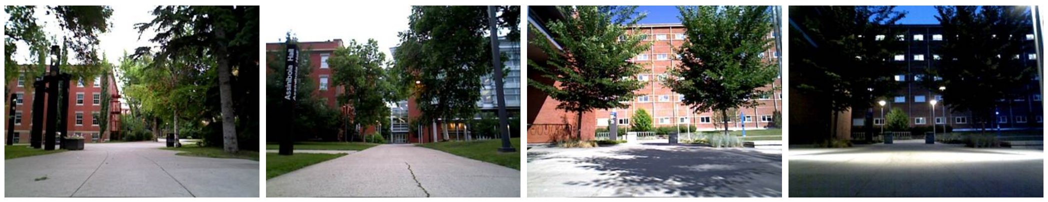

(a) (b) (c) (d)

Figure 1.2: Two examples of perceptual aliasing. (a) and (b) are different locations but look similar. (c) and (d) represent the same location but show different appearance due to illumination change.

partition of the feature space. Then two issues of 1) if the feature space used to train the vocabulary resembles the one in which feature descriptors are generated from robot views, and 2) if the feature space can be properly partitioned to distinguish the features, become the possible bottlenecks in achieving good performance in image comparison. Consequences of perceptual aliasing include a low recall in place recognition due to missed loop closures and a high cost in geometric verification for loop closure candidates that need to be processed.

Limitations in Feature Extraction and Representation

Many state-of-the-art algorithms in appearance SLAM that depend on local invariant features require keypoint extraction (both detection and description) and matching. Even if the time-consuming matching process can be alleviated by indexing visual words in BoW framework, the process of extracting keypoints is inevitable, and this is typically inefficient. Using binary visual words [35,36] is a more efficient alternative as both the detector (e.g., FAST [96]) and descriptor (e.g., BRIEF [12]) are fast to obtain. However, it does not radically solve the intrinsic perceptual aliasing problem mentioned above, especially in significant illumination changing environment. Another family of approaches use a whole-image descriptor which is much more efficient in encoding an image. The potential problem is that the temporal coherence in an image sequence is not exploited in many existing algorithms [76,109] and therefore they may not be scalable in large maps.

Difficulty in Geometric Verification

Multi-view geometric (MVG) verification is usually the last step in appearance SLAM. It is a highly desired procedure (although optional) as false positives are usually not allowed, i.e., 100% precision is desired in loop closure detection. MVG-based verification is expected to remove all the candidates that are not true loop closures and retain the real positive ones. It is only performed on a limited number (usually one or two) of top candidates with highest possibility of being a matching location, since this procedure is typically time-consuming. The fundamental step of verification is to build a set of feature correspondences between two images, i.e., perform reliable feature matching, which is extremely difficult under significant illumination change due to both degradation in feature repeatability and descriptor variance. A recent method is to exploit the vector field consensus [66] from putative matches. However, the algorithm is both too sensitive to parameters and inefficient due to the iterative *EM* procedure in estimating an implicit probabilistic model reflecting the flow pattern. As a result, a robust feature matching algorithm is important and should be developed in achieving reliable verification of loop closure candidates in order to obtain high accuracy in both precision and recall.

To summarize, several problems are present in the research of appearance SLAM in illumination changing environment, and need to be well addressed.

1.3 Motivation

Based on the above-mentioned problems in appearance SLAM, we conduct systematic research in this area, focusing on the challenging case of illumination change, with the intention of solving the problems of perceptual aliasing, feature limitation and inaccurate verification under illumination change.

First of all, as a fundamental problem in appearance SLAM, perceptual aliasing directly impacts the performance of robot localization and mapping. In dynamic environment with perceptual changes such as illumination variance, seasonal change, object motion and occlusion, etc., this problem becomes even more obvious and typical. A visual appearance SLAM system can neither reliably build a correct topological map nor achieve accurate localization with perceptual aliasing existing. Consequently, visual navigation is unavailable in such a case. Therefore, to ensure that a robot with visual sensors can be truly autonomous in complex unknown environments, perceptual aliasing is the urgent problem that needs to be properly addressed.

On the other hand, real-time implementation in robotics applications is required, i.e., all the tasks must be completed on-the-fly. In appearance SLAM, image encoding and comparison should be done within the time of key frame [129] sampling period (usually no more than a second). Many existing algorithms are limited by feature extraction and representation, and beyond real-time implementation. As a result, they are not really applicable in real-world scenario. In addition, developing a real-time SLAM system that is highly scalable in large maps [74] is the crucial step of the research towards its practicability. This can be addressed by improving the feature efficiency with scalable localization and mapping algorithms.

Finally, verification is the last insurance of accuracy in place recognition. A reliable feature matching method under illumination change enhances robustness of verification and improves detection performance. In our research, we study the feature matching method in the context of illumination change, incorporating it to the SLAM system.

1.4 Contributions

Motivated by the research problems and applications, this thesis studies appearance SLAM in illumination changing environment, using discriminative feature descriptors with efficient matching techniques to address perceptual aliasing problem. For challenging environment with significant illumination variance, we propose to use a whole-image descriptor to encode an image, plus an efficient localization and mapping algorithm that scales well in large maps. In addition, we develop a robust feature matching algorithm in order to achieve reliable verification. Particularly, the following contributions are made.

- We use original local invariant features, rather than their BoW representation to encode an image, and perform direct feature matching in image comparison to detect loop closures. Different from existing analogue methods in appearance SLAM that match features with nearest neighbor search, we propose to use a tree structure in organizing the pool of features from the map node images (key frames). Matching is then achieved by indexing visual features in relevant key frames and becomes much faster with superior performance to BoW-based image matching. In our method, loop closure detection can be achieved in real-time manner in a moderate environment. It also shows potentiality in illumination changing environment. In addition, to investigate the scalability of the method, we also apply the scale dependent feature selection [128] in our approach and show that online implementation is possible in large scale environment with satisfactory performance.

- In the case of significant illumination change and large environment, we use a compact whole-image descriptor, Garbo-Gist of low dimension to describe appearance and measure similarities among images. We employ PCA to transform a high dimensional Gabor-Gist descriptor to a lower dimensional form to improve both the computational efficiency of our method and the discriminating power of the image descriptor. In addition, we use Monte-Carlo sampling [28], particularly, a particle filter to exploit the correlation among images in a sequence captured by the robot in the process of identifying loop closure candidates, making our method highly scalable due to the compactness of the image descriptor and the simplicity of particle filtering. The algorithm is also extended to the case of matching image sequences, showing its superiority in efficiency and scalability.

- Performance evaluation of whole-image descriptors is provided. The study is conducted with several different types of whole-image descriptors in the context of SLAM and robot localization. As whole-image descriptors have been widely used in related research, it is interesting to compare the effectiveness of these descriptors and set up a guideline of choosing them in the context of loop closure detection or any other applications that can benefit from a compact and discriminative image descriptor.

- Assuming a weak perspective camera model which typically applies to outdoor robot navigation, and based on the study of spatial statistics of optical flow [97], we propose an reliable feature matching method for verification under significant illumination change. Our method uses a simple constraint derived on correctly matched keypoints to prune the putative matches to boost the inlier ratio significantly. With the assistance of the method, the performance of Loop closure verification is significantly improved.

In general, our research addresses several issues in appearance SLAM and provides solutions to the existing problems. By incorporating the methods proposed in this thesis, we aim to build a real-time SLAM system that can work in illumination changing environment by obtaining satisfactory localization and mapping performance. The achievements in our study have great potentialities and significance in robot applications.

1.5 Organization

The remaining of this thesis consists of five chapters. In Chapter 2, we introduce the efficient direct feature matching method in addressing perceptual aliasing, a common issue under illumination change. Inspired by both the widely used tree structure in approximate nearest neighbor matching and the inverted indexing in BoW, we propose the method of indexing visual feature in loop closure detection. Chapter 3 presents a novel localization and mapping framework using a compact whole-image descriptor and a Monte-Carlo implementation of Bayes filtering, with particular intentions on resolving the problems in illumination change and improving the scalability. In addition, the proposed method is extended to another case, combined with sequence-based image matching to show its possibility in improving the matching efficiency. A thorough performance evaluation of whole-image descriptors is conducted in Chapter 3, in the context of multiple robot applications such as SLAM, localization, kidnapped robot problem, etc.. The main purpose is to provide a general guideline of using these descriptors in different scenarios. A robust feature matching algorithm intended to improve the accuracy in loop closure verification under significant illumination variance is described in Chapter 4. The proposed method ensures that loop closure verification can be achieved reliably even in an extreme case of illumination change, such as between sunny day and dark night. Finally, we summarize the contributions in this thesis and present the future work in Chapter 6.

$\sim$

Chapter 2

Indexing Visual Features with a Tree Structure

In this Chapter, we will discuss the solution to perceptual aliasing by direct feature matching. As mentioned before, perceptual aliasing is a major issue in appearance SLAM, especially in changing illumination environment, and therefore should be properly addressed. The proposed method can be applied to real-time loop closure detection in an indoor environment. However, we show that it can also be possibly generalized to a larger environment by using feature selection to reduce the search space. In addition, it shows potentiality in appearance SLAM under illumination change. The core of the method is to use a *k*-d tree (or randomized *k*-d forests) to organize the features in the map images (key frames) so that approximate nearest search for a given feature descriptor in the robot view is possible. It is better than the traditional BoW methods as the true matching locations are always ranked high in terms of the number of matching features, and therefore saves time in the following MVG-based verification.

2.1 Introduction

Appearance SLAM using a BoW representation as the image descriptor dominates the early research in this topic, and is often considered the benchmark for comparison. As the introduction, we briefly recall some of the representative algorithms in the literature for this types of methods.

Newman *et al.* developed a SLAM system in [86] with multiple sensors. A geometric 3D map was built using a laser range finder with the aid of visual sensor to detect loop closures based on appearance recognition. Later, the BoW image representation was respectively used by Cummins *et al.* [23, 24] and Angeli *et al.* [1, 2]. In FAB-MAP [23], the authors used a Chow-Liu tree to capture the co-occurrence statistics of the visual words and proposed a method for explicitly calculating the normalizing term ($P(\mathbf{z}_k \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})$ in Equation (1.2)) in the recursive Bayes estimation for location likelihood. Note that in SLAM we have to consider the case of the current robot view being a new place, which is different from pure localization where the denominator can be simply computed by

$$P(\mathbf{z}_k \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}) = \sum_{i \in N} P(\mathbf{z}_k \mid \mathbf{x}_i, \mathbf{m}) P(\mathbf{x}_i, \mathbf{m} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0) \tag{2.1}$$

with N being the set of all the mapped places. Instead,

$$\begin{aligned} P(\mathbf{z}_k \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}) = &\sum_{i \in N} P(\mathbf{z}_k \mid \mathbf{x}_i, \mathbf{m}) P(\mathbf{x}_i, \mathbf{m} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0) \\ &+ \sum_{j \in \overline{N}} P(\mathbf{z}_k \mid \mathbf{x}_j, \mathbf{m}) P(\mathbf{x}_j, \mathbf{m} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0) \end{aligned} \tag{2.2}$$

where $\overline{N}$ is the set of unmapped places, and in [23] was achieved by sampling an arbitrary environment. The probabilistic framework was a success in challenging outdoor datasets in detecting loop closures. This model was modified in FAB-MAP 2.0 [24] in conjunction with a *k*-d tree used in the clustering process for generating many more visual words to make the system be able to deal with large outdoor environment.

Similarly, Angeli *et al.* also incorporated Bayesian framework to BoW representation to compute the likelihood of loop closure occurrence [1]. A "virtual image" was created at each likelihood computation step with the *n* most frequently seen visual words to account for the new location, assuming that loop closing images should contain unique visual words that can distinguish them from others. In [2], the authors used a thresholding method to avoid the update of loop closing hypotheses with too low weights, and hence simplify the computation of full posterior. In fact, only a restricted set of hypotheses need to be updated at every step. In addition, color information of local color histogram in the HSV color space was also added to improve the performance.

2.2 Feature Matching in SLAM

The major problem of perceptual aliasing can be intrinsic to the scene or artificially created by the process of image representation, for example, by clustering in creating the visual vocabulary in BoW. When a robot is building a map in an indoor environment such as a hallway, dense sampling of the images is required so that the the entire environment can be covered and the map is complete and meaningful for robot navigation. In such a case, there can be much overlap between consecutive images, and perceptual aliasing may appear more frequently than in a larger outdoor environment. A natural way to avoid this problem is to discard the quantization procedure and compare images using their raw local invariant features, for example, SIFT, by direct feature matching. In fact, feature matching has been extensively applied in vision community for object recognition and detection in various conditions [65]. In robotics, researchers have also tried using this method instead of BoW to achieve image comparison. The main concern however, is that as mapping is an incremental process, it can become computationally burdensome as the search space increases with the map growth. Therefore, how to speed up the search to find a match for a given feature becomes a crucial issue. Here we discuss the existing feature matching methods in SLAM and their possible speed-up extension using fast approximate nearest neighbor matching.

Zhang in [129] has shown that similarity using feature matching is a good criterion in comparing images, or equivalently, contend-based image retrieval (CBIR). The authors in [53] proposed a new kind of local feature named position-invariant robust feature (PIRF) to handle the problem. PIRF is in fact the stable local features that can be tracked in several continuous frames. It is the common case that along multiple (e.g. five) consecutive frames, only a few matching keypoints can preserve in all these images. Therefore, when these distinctive features appear again, it is the evidence that loop closure is detected. Apparently, the drawback is that a lot of feature matching needs to be done, not only in finding PIRF, but also matching them. A better solution in [128] is to use scale dependent feature selection [124] to remove a majority of SIFT keypoints extracted at fine scales and retain a small subset of the coarse scale features with higher repeatability.

Even though the number of features can be possibly reduced by feature selection, as map grows, still too many of them need processing, and matching by nearest neighbor search is intractable for a huge pool of features. An alternative is to use *k*-d tree in which features are assigned to the nodes in the tree and search complexity is reduced. Each dimension of the feature is used to evenly separate the data and the tree is in general balanced. A query feature reaches a leaf node in a few steps and a backtrack process using for example, best bin first (BBF) [7], based on the candidate leaf node is applied to find the approximate nearest neighbor of the query feature. Muja extended the tree-based search by automatically determining the best algorithm and parameter values [80, 82] given a dataset and desired degree of precision in search, and also applied the search in binary features [81]. In addition, the idea of a tree structure was used in clustering [87] to generate visual words as an improvement of [103]. The basic idea is to run *k*-means hierarchically with only a few cluster centers (visual words) at each level so that both clustering and search is much faster. Some applications of *k*-d tree (or forest) in feature matching in the vision community can be found in [90, 102].

Another efficient data structure in approximate nearest neighbor match is locality-sensitive hashing (LSH). The idea of LSH is to find a set of hash functions so that similar vectors are likely to have the same hash value. There are different hash function families with different distance metrics and a commonly used example is the one proposed by Datar [26] that optimizes Euclidean distance. A hash value is created by projecting the original feature vector to a random direction and shifting the projection length by a random amount. One advantage of LSH is that it does not need any preprocessing such as building the tree. As long as the hash functions are defined, new features can be added to the hash tables for indexing. However, it is shown in [82] that *k*-d tree has better performance. Examples of using LSH in image retrieval are shown in [20, 21] and there have also been applications in robotics [8, 99] for detecting loop closures.

In a word, our feature matching method is highly inspired by the above-mentioned algorithms and applications in both vision and robotics community. The key issue is to use an efficient data structure to make feature matching possible in a huge search space so that robust image comparison can be achieved to overcome the perceptual aliasing problem in the traditional BoW approach.

2.3 Appearance SLAM by Feature Matching Using *k*-d Tree

Compared with the existing work on visual loop closure detection where *k*-d tree is used in clustering, our method uses *k*-d tree to detect loop closures via feature matching and no vector quantization is used. The idea is inspired by several observations.

We first notice that *k*-d tree not only reduces the search complexity from linear to logarithmic, but also compares one dimension of the high-dimensional features each time and therefore avoids the distance computations, the most time-consuming part in finding the correct nearest neighbor for a query feature with linear search. The previous work such as [24] used *k*-d tree (or randomized *k*-d forest) to assign each feature to its closest cluster center during the clustering process. This could make the clustering possible with a large amount of features to generate tens of thousands of visual words. However, the perceptual aliasing problem may deteriorate by erroneously classifying the features to their corresponding cluster centers, although we have no way to validate how significantly this approximate *k*-means clustering [87] may impact the performance of loop closure detection since standard *k*-means over too large dataset is intractable and no comparison can be made then. Nonetheless, *k*-d tree can still be a strong and efficient tool for approximate nearest neighbor search, which is an essential part of our method. In fact, people have shown the possibility of using *k*-d tree or randomized *k*-d forest in matching high-dimensional image descriptors [80, 82, 102].

The second observation is that the distance ratio method [65] is an effective way for verifying putative matches. By using the distance ratio test, a correct match requires the ratio between the distances of the closest and second closest neighbor to the query feature to be below some given threshold. This implies that we can have a way of handling incorrect nearest neighbors returned by a *k*-d tree. In our work, we focus more on the applicability of the tree structure in feature matching by using distance ratio technique.

Some other phenomena that can make *k*-d tree suitable to our work will be discussed later as these observations can be explained clearly together with the proposed algorithm.

2.3.1 Key Frame Selection and Construction of *k*-d Tree

In order to test the performance of loop closure detection, we first need to select a set of key frames for comparison. Key frames are representative images taken from distinct places of the environment. As discussed in Section 2.2, images are usually sampled densely in an indoor environment in order to ensure a full coverage and therefore considerable overlap may exist along a set of continuous frames. In such a case, there is no need to keep all the captured images since many consecutive ones look similar and a subset of sampled images are enough. There may be several ways of selecting key frames, for example, at a certain fixed time interval (e.g. per second) as the robot camera takes images. While this simple method works well for the case of a straight robot trajectory with a constant linear speed, it may miss key frames in the case of abrupt change of the location appearance, for example, when the robot makes a turn at a corner. Here we use the key frame extraction strategy proposed in [129] with some modifications to select key frames. Particularly, the first image captured is always considered as the first key frame and two consecutive key frames should either have little overlap or be "far apart" enough spatially, whichever occurs first,

$$overlap = \frac{2N}{N_i + N_t} \tag{2.3}$$

$$distance = |ID_i - ID_t| \tag{2.4}$$

where N_i and N_t are the numbers of scale-invariant features (we use SIFT in our work) in image I_i and I_t respectively. N is the number of matching features while ID_i and ID_t are the sequential numbers of I_i and I_t showing the order of sampling. For Equation (2.4) to be valid, we assume constant velocity in robot motion.

Thresholds are given to the above equations for deciding whether the current view should be considered as a new key frame.

Once a new key frame is detected, we insert its features to the feature pool D and construct a *k*-d tree over all the features in D. Hence, D always includes the features extracted from all the key frames and the tree is updated every time a new key frame is found. Inspired by the inverted index used in BoW, which keeps the associated image *id*s and word frequencies for each visual word, we also maintain a similar table which maps the features in D to their corresponding images from which they are extracted.

2.3.2 Indexing Visual Features

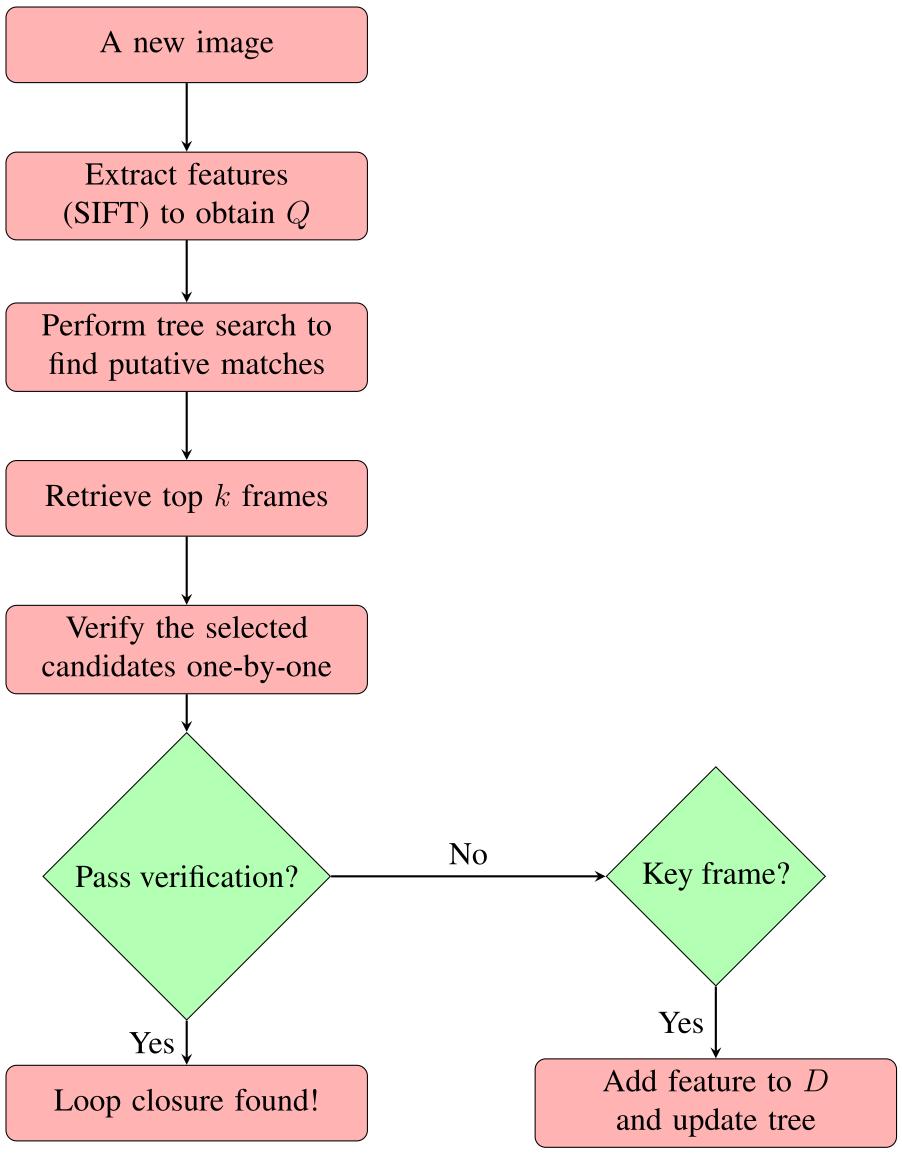
Let Q be the set of features extracted from the current image. For each feature in Q, it will go through a tree search in the current *k*-d tree and the top two nearest neighbors are returned. Distance ratio is applied to determine whether the closest one is a good match or not. If positive, the image containing this matching feature can be immediately retrieved by indexing.

Since *k*-d tree does not guarantee to find the correct nearest neighbors, the key frame that contains the most matching features is not necessarily the true loop closure key frame. We then perform a second step of further verification of the possible candidates. In this step, top K key frames that contain the most matching features are selected as the candidates and direct feature matching is performed between the query image and each candidate. A loop closure is found if the similarity is above some given threshold. This verification is based on the observation that only a few key frames contain matching features after the previous indexing step and the true loop closure key frame is always likely to have multiple matching features by using the tree search. One can also use a more rigorous and accurate MVG-based verification that identifies the underlying camera motion between two frames based on epipolar constraint (we will discuss it in detail later in Chapter 5), at the expense of spending extra time on this step.

We also notice that the features in Q that have correct matches in D only represent a small fraction (normally less than 15%) of the total features in Q. This may potentially increase the tolerance of finding the incorrect nearest neighbors. The other important fact is that we are only interested in finding the correct matches for features in Q that do have matches in D. In other words, the purpose is to find as many true positive samples as possible using the tree search and the number of false positives is not important. The only possible negative impact of finding a false positive is to retrieve an irrelevant key frame as a candidate after the first step of the algorithm. However, this irrelevant key frame is likely to be discarded since it is unlikely to rank high among all the key frames with matching features.

2.3.3 Loop Closure Detection and Simple SLAM System

The algorithm is summarized in Algorithm 2.1 with a brief flow chart shown below. Here we use $|A|$ to represent the number of elements in set A and variables that are not mentioned in the algorithm have the same meaning as in Equation (2.3) and (2.4). The similarity measurement is consistent with Equation (2.3). In Algorithm 2.1, D is initialized to be an empty set and increases with features from key frames. q is the current image. E is the set of key frames (we can use their *id*s) that contain matching features and C is the sorted top K candidates in E according to the number of matching features (if $|E| < K$, we will consider all the key frames), returned by the function sorttop(K, E). DIST and SIMI are two thresholds used for discarding nearby key frames to the current view and determining loop closures. In step 1, a set (could be empty) of ranked key frames based on the number of matching features using tree search are generated, and step 2 performs feature matching to make a loop closure decision. If no loop closure is found, the robot needs to add the features to D if the current observation is a key frame, and update the tree. Note that the nearby key frames are not loop closures but may have many matching features due to the short spatial distance to the current view. The algorithm takes Q, a set of SIFT feature descriptors as the input, and returns a loop closure decision and the associated key frame *id* if existing. It is developed to detect loop closures, but can be easily incorporated to a SLAM system, by making topological connections among the key frames and updating the map with new locations to fulfill the state augmentation in


A new image
Extract features (SIFT) to obtain Q
Perform tree search to find putative matches
Retrieve top k frames
Verify the selected candidates one-by-one
Pass verification?
No
Key frame?
Yes
Loop closure found!
Yes
Add feature to D and update tree

SLAM.

2.4 Experimental Results

In order to validate the performance of the proposed method in visual loop closure detection and appearance SLAM, several groups of experiments were conducted on both a dataset describing an indoor hallway environment, and another one collecting from an outdoor campus environment incorporating significant illumination change. In this section, we will illustrate the results from both environments respectively.

Algorithm 2.1: Visual Loop Closure Detection Using *k*-d Tree

```
while robot acquiring images do
    % step 1: indexing visual features %
    $E = \text{matchTree}(Q, D)$ ;                          // tree search
    for $i = 1 \rightarrow |E|$ do
        if $\left|ID_q - ID_{E(i)}\right| \leq \text{DIST}$ then
            $E = E \setminus E(i)$;
        end
    end
    $K = \min(K, |E|)$;
    $C = \text{sorttop}(K, E)$ ;   // sort the top K candidates in E
    % step2: further verification in image level %
    $i \leftarrow 1$;
    while $i \leq K$ do
        $N = \text{matchIm}(q, C(i))$ ;                 // feature matching
        $S = \frac{2N}{N_q + N_{C(i)}}$;
        if $S > \text{SIMI}$ then
            a loop closure is found;
            break;
        end
        $i \leftarrow i + 1$;
    end
    % key frame detection if needed %
    if $i > K$ then
        if $\text{keyframeDetect}(q) = \text{TRUE}$ then
            $D = D \cup Q$;
            $\text{updateTree}(D)$;
        end
    end
end
```

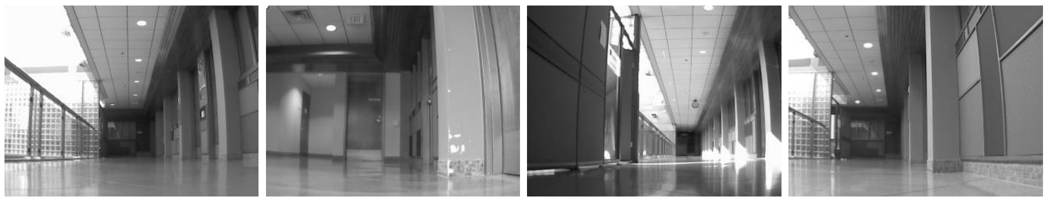


Figure 2.1: Sample images of the indoor dataset. The dataset was sampled densely and covers two entire loops of the hallway in the second floor of Computing Science Center, University of Alberta.

2.4.1 Indoor Environment

The indoor dataset was sampled by a Pioneer 3-AT mobile robot equiped with a Dragonfly IEEE-1394 camera pointing forward [128]. Images were captured at equal time interval as the robot moved, covering a total distance of approximately 200 meters with two entire loops of the second floor of Computing Science Center, University of Alberta. A total of 7420 images were collected with resolution of 320×240 pixels. Figure 2.1 shows several sample images of the dataset and Figure 2.2 is the visualization of the path (unit: meter) according to the odometry readings. Due to the error in the readings from robot wheel odometry, the original trajectory of the two loops was not quite aligned. We performed an easy calibration of the sensor so that the visualization looks better as shown. Note that no metric information of any type was used in our method.

Performance of Feature Matching Using *k*-d tree

Before applying the proposed method to loop closure detection, we would like to first investigate the possibility of using *k*-d tree in feature matching in terms of both accuracy and run time. We extracted all the SIFT features from the above dataset and randomly selected 500,000 of them as the feature pool D. Another random 20,000 features from the same dataset were used as the test query set Q. By performing a linear search for each query feature over all the features in D, we obtained the ground truth of the test set and 2415 features out of 20,000 (around 12.1%) had been given corresponding matches. FLANN was used as our implementation of *k*-d tree [80]. Different numbers of trees and checked leaf nodes were used in the tree

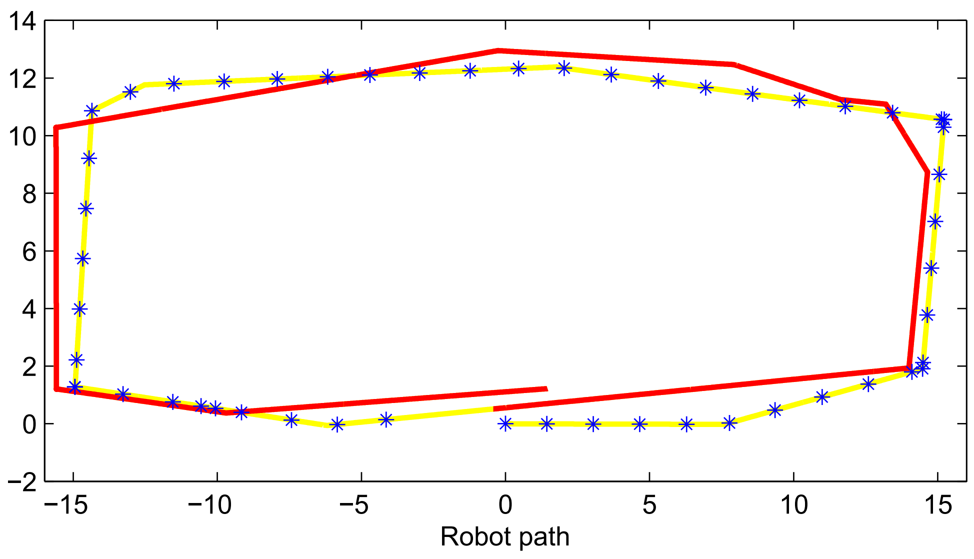


Figure 2.2: Visualization of the data describing robot path and key frames according to the odometry readings (after alignment). There are two complete loops. 3346 images taken in the second loop (red) are considered loop closures. Selected key frames in the first loop (yellow) are shown in blue stars. The starting point is [0, 0]

search to illustrate the performance. As discussed previously, we are only curious about how many correct matches we can find for these 2415 features. Figure 2.3 shows that the performance will improve by using multiple trees and checking more leaf nodes in the backtrack process. There is no obvious difference between three and four trees and this suggests that the performance may converge to a number of trees. Hence, it is unnecessary to use too many trees in our method. Note that the search time will also increase if more trees are used or more leaf nodes are examined. In fact, we used only one tree and compared 10 leaf nodes in the experiments on this dataset and found it working pretty well. As shown in the figure, we are still able to get around 90% of matches correct, which is an acceptable result.

Another group of experiments was designed to investigate the applicability of *k*-d tree in terms of run time. Since the number of key frames is increasing during SLAM process, we are particularly interested in seeing how the speedup factor over

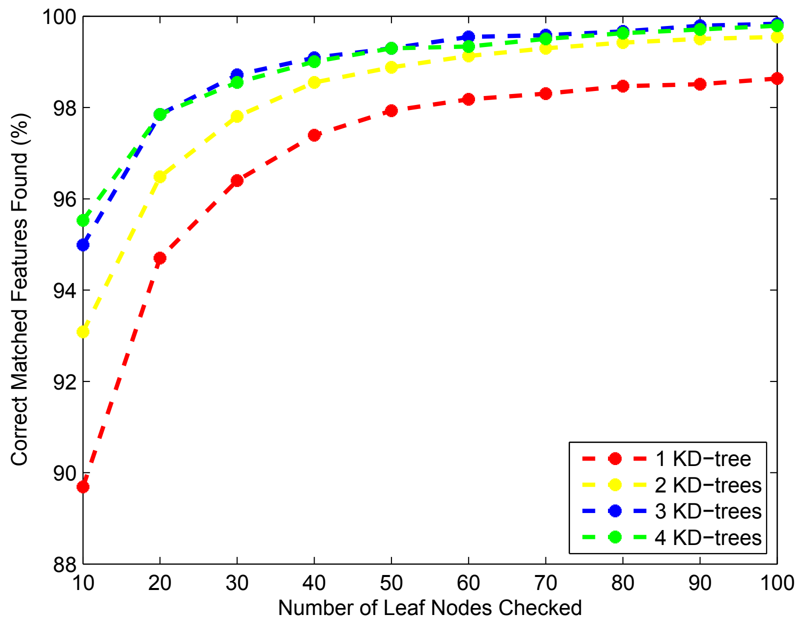


Figure 2.3: Result of feature matching with different numbers of trees and checked leaf nodes. The result suggests that in terms of accuracy, *k*-d tree can be used in loop closure detection. There is no need to use multiple trees since the accuracy is good enough with one tree for our problem.

linear search evolves with the increase of feature number. Different numbers of features were randomly collected from the same dataset, ranging from 10K to 100K as D. The same test set Q was used as the query samples. There is no overlap between any of the two feature groups (including the previous 500,000 feature group). The second row of Table 2.1 shows the numbers of matching features between the test set and different pools using exhaustive search as the ground truth. The tree search performance (here one *k*-d tree was used with 10 leaf nodes checked) in the third row is consistent to the result shown in Figure 2.3. Figure 2.4 is the speedup factor of *k*-d tree over linear search on different feature groups. In the tree search, we counted the time of both constructing the tree and performing the search. The result shows that *k*-d tree is at least three orders of magnitude faster than linear search and the time advantage increases rapidly with the increase number of features.

Table 2.1: Number of Matching Features and Performance of Tree Search in Different Feature Groups. The second row shows the number of true matches (ground truth) in each group. The third row shows the correct matches using *k*-d tree in percentage.

10K	20K	30K	40K	50K	60K	70K	80K	90K	100K
1747	2251	2376	2529	2615	2609	2819	2766	2760	2688
88.61	86.94	86.15	87.90	88.83	87.93	86.91	87.20	86.81	87.69

Loop Closure Detection

Based on the above results, we ran the proposed loop closure detection algorithm described in Algorithm 2.1 on this indoor dataset on a normal lab computer (2.0 GHz CPU and 4.0 GB memory) and compared the results with BoW of 1000 visual words in terms of both recall and time (the visual words were generated by clustering more than 20000 features extracted from sampled images of another dataset). The scale dependent feature selection proposed in [128] was also applied here to show the scalability of the method. Selected features were used to build the tree (features in D) and search (features in Q). The run time of the algorithm mainly comes from three aspects: time of building the tree, search time (indexing in step 1) and time of further verification (step 2). All the other components such as feature extraction and key frame detection are the same to both algorithms and do not need to be considered and compared. Thresholds for Equation (2.3) and (2.4) are 3% and 80 respectively. DIST and SIMI are 100 and 3%, i.e., we do not want to consider the key frames within 100 images in sequential order as the loop closure candidates, and the threshold for accepting a loop closure hypothesis is the same as deciding a key frame. The parameters were all empirically chosen and could be adjustable. The guideline is that the key frames should have a full coverage of the environment but are not redundant in localization. Different groups of parameters were tried and the differences between the proposed method and BoW on different parameter settings were close. A total of 53 key frames were selected (see Figure 2.2). We built the ground truth mainly according to the spatial distances: for each image in the second loop, the closest two key frames are both considered the true

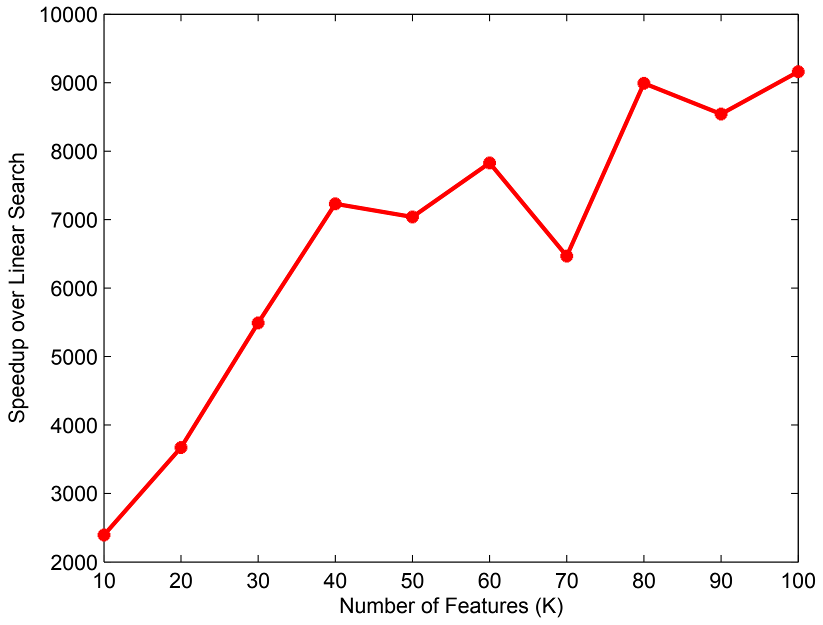


Figure 2.4: Speedup factor over linear search vs. different numbers of features. *k*-d tree is at least three orders of magnitude faster than linear search. It is reasonable to believe that we are able to deal with many key frames in real-time implementation using *k*-d tree in feature matching.

loop closure images. This would be subject to a visual check to exclude special cases (e.g., when the robot turns, the two images may look different although they are spatially close). The performance is summarized in Table 2.2 for the 3346 loop closure images in the second loop in terms of recall. As mentioned in Section 2.3.3, we consider the top K ranked candidates in E to undergo the further verification in step 2. Here we do not worry about the precision because step 2 will reject any candidate that does not have a similarity measurement above the given threshold. In other words, with this step we can consider the precision as 100%. The first column in Table 2.2 is the scales above which features were used [128]. So "> 0" means that the entire original SIFT features were used without applying scale selection. It can be seen that the performance will go down with fewer feature used (more feature selection applied). This is understandable because the number of features is too few. However, our proposed approach can be still superior to BoW even if we reduce the average number of features to only 30 per image (scale > 4). The top 3 and top 5 results do not differ too much which suggests that the correct loop

Table 2.2: Recall on Indoor Dataset with Scale Selection (%)

	Top 1	Top 3	Top 5	# of features / image
> 0 (all)	85.12	94.77	95.70	256.6
> 1	82.46	94.17	95.31	211.1
> 2	80.66	91.54	92.17	78.9
> 3	78.81	88.52	88.82	46.4
> 4	72.33	82.85	82.93	30.8
BoW	45.04	72.33	80.39	256.6

Table 2.3: Run Time on Indoor Dataset – Building the Tree (ms)

	Min	Max	Average
> 0 (all)	2.1	31.2	16.2
> 1	1.5	26.4	13.6
> 2	0.89	31.2	5.8
> 3	0.66	24.8	3.7
> 4	0.56	26.0	2.8

closure key frames are normally ranked high, with or without feature selection. The run time of both building the tree and indexing is shown in Table 2.3 and 2.4 respectively. By using features at scales higher than 4, the average run time is almost one order of magnitude less compared with using the entire set of original features, sacrificing the performance as much as 13%. In most cases, the average time of processing one image is the total time of both search (step 1) and further verification (step 2). The time of rebuilding the *k*-d tree should be counted in only when a new key frame is found. Therefore our method provides a real-time performance to detect loop closures on this dataset.

To make the results comparable, we also used a single *k*-d tree with 10 leaf nodes in the backtrack to create the BoW descriptor of an image from the extracted features. Therefore, the time of BoW in Table 2.4 is actually the time of converting the image to the BoW descriptor, depending on the vocabulary size and the number of features in the query image. For the top K retrieved candidates with the highest similarity according to their BoW descriptors, the same verification was performed

Table 2.4: Run Time on Indoor Dataset – Search and Indexing (ms)

	Min	Max	Average
> 0 (all)	0.55	128.4	10.3
> 1	0.40	97.9	9.0
> 2	0.30	50.8	4.9
> 3	0.27	25.4	3.3
> 4	0.24	16.7	2.3
BoW	0.32	3.7	1.4

to decide whether a loop closure was found or not. The combined run time of both indexing and verification can be represented as

$$T = T_i + k \times T_v \tag{2.5}$$

where T_i is the indexing time shown in Table 2.4 and T_v is the time of further verification for each retrieved candidate. k is the number of candidates verified to achieve a certain level of performance. We are only interested in comparing T because all the other parts are the same to both methods as mentioned previously, and the time of verification can be dominant in the total execution time of processing one image, if too many candidates are verified. The verification for each key frame took around 6.9 milliseconds on average. Fig 2.5 shows T in both methods with respecct to the performance. It is not surprising to see that BoW spends more time in achieving the same level of performance. In the proposed method, the loop closure key frame is usually included in the top 5 retrieved candidates after the indexing step. However, BoW needs to verify as many as tens of key frames to obtain the correct one. This verification is mainly responsible for the time spent in detecting loop closures.

The series of experiments designed above have obviously shown the effectiveness and efficiency of the proposed method in visual loop closure detection and appearance SLAM. The results have shown several facts. First of all, direct feature matching in appearance SLAM is possible with fast approximate nearest neighbor search using *k*-d tree. In addition, perceptual aliasing indeed exists in the state-of-the-art BoW method and consequently, a low recall can be generated due to missed

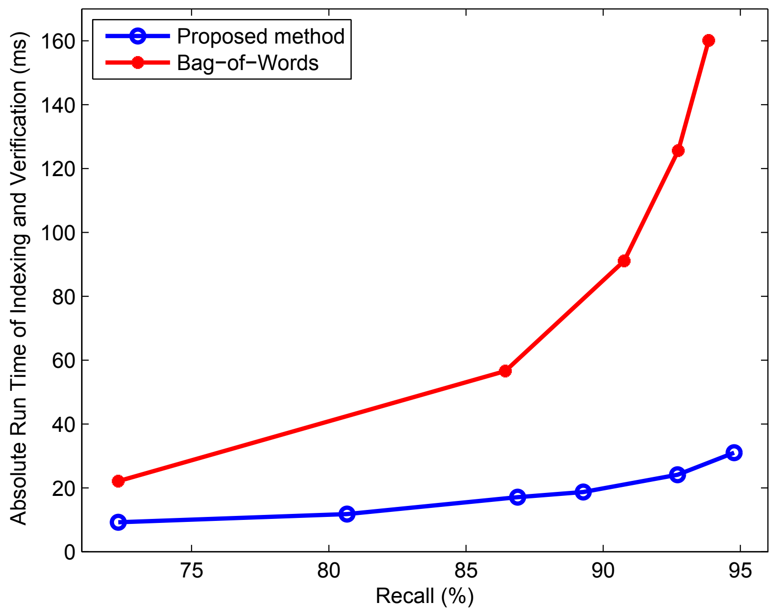
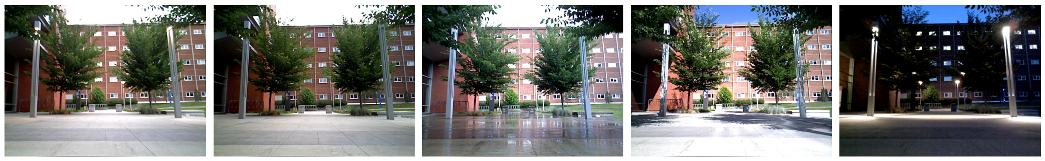


Figure 2.5: The absolute run time of indexing and verification vs. recall of both methods. Since BoW requires more key frame candidates to be verified to achieve the same level of performance, it is not as efficient as the proposed method on this indoor dataset in terms of obtaining the same recall.

loop closures. In order to overcome this issue, a brute-force solution is to verify more candidates. This is apparently not desirable due to the time-consuming process of verification. Our method alleviates the problem as the true loop closure key frame is always ranked high. Finally, scale dependent feature selection can help improve the efficiency without loss of too much performance. This increases the scalability of our method.

Based on the experimental results shown above, we will then conduct another group of experiments, focusing on the case of illumination change using the proposed method. As most of our hypotheses have been validated in the previous experiments, the following experiments will be designed and conducted in a brief and compact way, showing only the performance of the method in a different environment.



(a) Cloudy1 (b) Cloudy2 (c) Rainy (d) Sunny (e) Night

Figure 2.6: Sample images of the outdoor dataset at the same location in five sequence, cloudy (two cases), rainy, sunny after the rain and dark night at 10 o'clock.

2.4.2 Outdoor Environment

In this set of experiments, we used a dataset in changing illumination environments. The dataset was collected with five image sequences at five different times of a day in a campus environment. The images were collected with a Husky A200 mobile robot equipped with a Xtion Pro camera. In each sequence the robot was driven along the same trajectory of approximately 700 meters at the speed of 1*m/s*. Therefore, the image numbers of each sequence can be used in building the ground truth of positive loop closures. Several weather and illumination conditions are covered in this dataset, including cloudy, sunny, rainy and night. Figure 2.6 shows a set of matching images in the five sequences representing the same location. In our experiment, we investigated the most challenging case of localization between dark night and sunny day. The images in the sunny sequence were used as a map (first loop) and every frame is a key frame (645 in total). The 646 images in the other sequence were compared with the map nodes.

Experimental settings are generally consistent with that in the previous indoor experiment. However, as the outdoor dataset contains high resolution color images with several thousand SIFT keypoints per image, we no longer considered using the entire set of descriptors as the execution is far beyond real-time in such a case. In the tree search, we still used one tree but 128 checks (rather than 10) in the backtrack to improve the performance in search. In addition, BoW was omitted since it has been validated to be inferior, by both our previous results in the indoor environment and the results reported in [76] where FAB-MAP with BoW achieves recall values lower than 10% in a similar environment in terms of the level of illumination variance. The performance is summarized in Table 2.5. As can be seen, the

Table 2.5: Recall on Outdoor Dataset with Scale Selection (%)

	Top 1	Top 3	Top 5	# of features / image	# of matches
> 1	70.90	76.16	78.17	2052.9	14.5
> 2	64.55	75.54	79.10	430.4	7.9
> 3	56.97	71.52	73.68	208.5	4.8
> 4	47.99	60.37	61.92	124.8	3.4

Table 2.6: Run Time on Outdoor Dataset – Tree Build and Search (s)

	Build	Search
> 1	16.34	0.19
> 2	1.67	0.035
> 3	0.78	0.016
> 4	0.45	0.009

recall is significantly lower than that in the indoor experiments, even if the number of features per image is much higher. The degradation in the performance shows the difficulty of appearance SLAM using feature matching in an environment with illumination change. In spite of the fact that the number of keypoints is high, their matching repeatability decreases as the illumination changes, and consequently, the number of true matches becomes fewer. The last column of Table 2.5 illustrates the average number of matches that can be found by the tree search in the top 1 case (the best matching candidate). With only a few matches, it is not confident enough to make a decision of a true loop closure.

The time of building a *k*-d tree and search will also become intractable with the increase of features. Table 2.6 shows the time of building a tree with 645 frames in the sunny sequence, and the average time of tree search for the 646 images in the night sequence. The search time is still acceptable even if the tree contains millions of nodes. However, building such a tree can be especially time-consuming. Note that the time complexity of building a *k*-d tree is $O(kn\log n)$, and the number of features (n) extracted at scales higher than 1 can reach several million. In such a case, building a *k*-d tree takes 16 seconds, no longer applicable to robot applications.

Although the experimental results of the proposed method deteriorate in the outdoor environment, most values are still higher than the state-of-the-art SeqSLAM [76] in handling significant illumination change with 60% recall. The method is even better than BoW in the less challenging indoor environment. Therefore, we can conclude from the results that 1) our proposed method of appearance SLAM or visual loop closure detection significantly outperforms BoW in addressing the problem of perceptual aliasing, and can be used in handling illumination change in a moderate size environment and 2) there are still improvement space and it is reasonable to believe that the matching performance can be improved if the accuracy of feature matching increases.

2.5 Summary

In this chapter, we have presented a loop closure detection method for appearance SLAM. Our proposed method does not rely on the widely used BoW representation to extract image descriptors but uses feature matching to address the perceptual aliasing and improve the recall in detecting loop closures. Our method consists of several steps. First, we built a *k*-d tree over all the database features extracted from the map images or key frames. For a newly acquired image, we also extracted its local invariant features and performed a tree search for these keypoints. Relevant matching candidates can be retrieved by indexing visual features to achieve fast match. In addition, we applied a further verification to confirm a true loop closure. It is shown in our work that due to its efficiency in feature matching, *k*-d tree can be applied in real-time loop closure detection with high recall in an indoor environment. The necessity of using scale dependent feature selection was also discussed for the purpose of extending the work to large outdoor environment.

In addition to the experiments conducted on a moderate size indoor environment confirming our method in terms of both recall and time, we also applied the method to an outdoor environment with illumination change. The performance decrease in both recall and time in outdoor environment has demonstrated the difficulty of SLAM in illumination changing environment. However, our method is still able to achieve up to 80% recall, showing its potentiality in dealing with illumination change. Possible improvement may involve boosting the accuracy in feature matching, which may be important future work.

One interesting issue is to discover and investigate more efficient feature selection strategy to keep the performance as high as possible while reducing the feature number. Inserting the features in the current tree instead of rebuilding it entirely every time is also an alternative to reduce the time. In addition, the framework of implementing SLAM in the proposed method is purely based on image comparison and does not take into account the filtering process. Therefore, map frames are considered random images without continuity. In appearance SLAM however, images are usually acquired sequentially and temporal coherence commonly exists along the robot trajectory (similar case also applies to traditional metric SLAM). In Chapter 3, we will exploit this important information to improve the performance in appearance SLAM.

$\sim$

Chapter 3

Appearance SLAM with a Compact Image Descriptor

In this chapter, we will present a method for SLAM using a compact image descriptor, Gabor-Gist. The traditional feature-based algorithms, including both BoW and feature matching mentioned in the last chapter, depend on the local invariant features in image comparison. As shown before, the performance of this type of methods may deteriorate when the illumination changes significantly. In contrast to these approaches, we develop a method relying on a single efficient image descriptor of low dimension to describe appearance and measure similarities among images. We employ PCA to transform a high dimensional Gabor-Gist descriptor to a lower dimensional form to improve both the computational efficiency of our method and the discriminating power of the image descriptor. To define SLAM in a probabilistic framework, we use a particle filter to exploit the correlation among images in a sequence captured by the robot in the process of identifying loop closure candidates. Our method is highly scalable due to the compactness of the image descriptor and the simplicity of particle filtering. In addition, it can be also extended improve the efficiency of sequence-based SLAM so that the issue of illumination change can be well addressed.

3.1 Introduction

In Chapter 2, we have presented the traditional solutions to appearance SLAM with some representatives using BoW as the image descriptor in a Bayes framework. The method presented in this Chapter uses a compact image descriptor of Gabor-Gist. Since no BoW or feature matching is used, neither the offline process of building the vocabulary nor the online extraction of any local invariant features (such as SIFT or SURF) is needed. Instead, a global image descriptor which is a compact representation of an image is used here. Comparison between images also becomes straightforward, using only the simple Euclidean distance. To eliminate the unnecessary comparisons with all the existing map nodes and reduce the computational cost, we exploit the temporal coherence in the image sequence with the particle filter framework to track the matching candidates. Note that particle filter is an implementation of Bayes filtering in probabilistic SLAM and ensures the continuity in detection so that temporal false detections can be removed. Compared with the existing work, our method uses an efficient likelihood function in a probabilistic framework and maintains a fraction of all the hypotheses during the loop closure detection process. Although we use Gabor-Gist as our image descriptor here, any other compact image descriptor can be an alternative. From this point of view, our method can be also considered as a general framework of loop closure detection or appearance SLAM.

3.2 SLAM with a Whole-Image Descriptor

In contrast to local invariant features, a whole-image descriptor is defined as a global and compact representation of an image, typically using a vector with either real numbers or binary values. This type of image encoding has been studied extensively in image-based robot localization and mapping.

Before the application of BoW to robotics, Ulrich and Nourbakhsh used color histogram as the features to describe an image, combined with a voting scheme to perform image matching [121]. In this method, the color histograms of training (map) panoramic images were built offline using HSL (hue, lightness, saturation) and normalized RGB color spaces. Histogram comparison using nearest neighbor matching was applied to each color band separately to generate the location votes of matching candidates and classification was done by unanimous voting. Apparently, the problem was defined as pure localization rather than SLAM. Badino *et al.* [3] described a topometric localization method which combines the robustness of topological localization with the geometric accuracy of metric methods. In image comparison of the topological part, the authors used a novel whole-image descriptor named whole-image SURF (WI-SURF), containing gradient information of the entire image, computed in the way of deriving a SURF keypoint descriptor. A filtering framework was also exploited in localization. Similarly in [109], a BRIEF descriptor [12] was extracted for the entire image with the center as the only "virtual" keypoint. The WI-SURF and BRIEF-Gist are both whole-image descriptors since they represent an image without keypoint detection. The binary whole-image descriptor has shown comparable discriminating power with real-valued descriptors, although they have not been exploited within a probabilistic framework.

Siagian and Itti proposed a biologically-inspired approach to scene classification [100] using a gist descriptor [88], which represents an image in terms of its responses to a filter bank such as discrete cosine transform or Gabor, and applied it to robot localization [101]. Principal Component Analysis (PCA) was performed on the original extracted gist features to reduce the dimensionality. In robot localization, a map was divided into segments – each of multiple locations – and a segment was described by the gist descriptors of the images captured by the robot at the locations in this segment. A neural network was trained for computing the likelihood of an image belonging to a segment. Subsequent salient region matching was employed to refine localization with the coarse localization hypothesis determined by the gist-based segment matching. Monte Carlo localization (MCL) that utilizes sampling importance resampling (SIR) [28, 34] was used as the back-end to estimate robot position.

More recently, Murillo *et al.* investigated the possibility of using the Gabor-Gist descriptor in visual loop closure detection with panoramic images of four views [84]. A Gabor-Gist image descriptor captures the image responses to a bank of Gabor filters. A Gabor filter is Gaussian kernel modulated by a sinusoidal plane wave and used to detect the edges in an image. It has the form

$$
\begin{aligned}
g(x, y) =& \text{exp}\left(-\frac{(x\text{cos}\theta + y\text{sin}\theta)^2 + \gamma^2(-x\text{cos}\theta + y\text{sin}\theta)^2}{2\sigma^2}\right) \\
&\times \text{cos}\left(2\pi\frac{x\text{cos}\theta + y\text{sin}\theta}{\lambda} + \psi\right)
\end{aligned}
\tag{3.1}
$$

In the above equation, λ represents the wavelength of the sinusoidal factor, θ represents the orientation of the normal to the parallel stripes of a Gabor function, ψ is the phase offset, σ is the standard deviation of the Gaussian envelope and γ is the spatial aspect ratio, and specifies the ellipticity of the support of the Gabor function. To obtain a Gabor-Gist descriptor, one can simply convolve an input image with different Gabor filters with different frequencies and scales, and use average intensities of the output image as the filter responses. In [84], when comparing two panoramas, the permutation of circular shifts of the four views was used to obtain four measurements, out of which the most similar one was adopted. In addition, the authors also proposed to use the gist vocabulary which consists of visual gist words to measure the similarity between two images. Under such a representation, the distance between two sets of original gist descriptors is equivalent to that between their corresponding gist words. PCA was also exploited to reduce the dimensionality and improve the matching performance. In terms of Bayes filtering in localization, a similar model as in [2] was used. The use of panoramas can be possibly applicable to the case of loop closure from opposite directions. It was shown that localization in a large area (more than 12K images covering a long run of 13 miles in urban area) is possible with gist, and the performance in place recognition is comparable or better than local feature-based approaches, with the advantage of higher efficiency and smaller memory storage requirements.

Some other applications of whole-image descriptors can be found in [73,74,76, 126]. In these methods, the image was downsampled to a smaller size and used as a descriptor. The descriptor values can be either the normalized pixel intensities or their binarization. The most obvious advantage is that this type of descriptors are even more compact and memory efficient. Therefore, mapping a much larger area becomes possible. In addition, combined with a framework of sequence-based matching, the simple whole-image descriptors can handle the case of significant illumination change [76], and drastically outperforms the traditional BoW image descriptor in such a case. However, only the discriminating performance was evaluated in these methods. The Bayes filtering in localization is not incorporated.

In general, whole-image descriptors have shown great potentiality in vision-based robot localization and mapping, including the case of illumination change, where BoW may not work well. There are two crucial components in the existing algorithms – the representation or selection of the descriptor itself and the implementation of the filtering framework in achieving robust localization performance. Inspired by the current methods that Gabor-Gist is a good choice in scene recognition and has shown success in robot localization, we use it in our proposed algorithm, plus a novel application of Monte Carlo Localization to achieve localization and mapping.

3.3 Monte Carlo Loop Closure Detection

To describe our proposed method, we first recall the probabilistic model of SLAM in Equation (1.1) and (1.2) in Chapter 1. To facilitate our representation, we use a simplified version of definition, ignoring the landmark state of $\mathbf{m}$ as in appearance SLAM, no explicit landmarks are estimated. Then the two equations can be represented as

$$\begin{aligned} P(\mathbf{x}_k \mid \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}) &= \frac{P(\mathbf{z}_k \mid \mathbf{x}_k) P(\mathbf{x}_k \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})}{P(\mathbf{z}_k \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})} \\ &= \eta P(\mathbf{z}_k \mid \mathbf{x}_k) \int P(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) P(\mathbf{x}_{k-1} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}) \, \mathrm{d}\mathbf{x}_{k-1} \\ &\propto P(\mathbf{z}_k \mid \mathbf{x}_k) P(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) P(\mathbf{x}_{k-1} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}) \end{aligned} \tag{3.2}$$

In the above equation, the denominator $P(\mathbf{z}_k \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})$ is considered to be a constant in our method. In topological SLAM, the map is represented by a graph whose nodes are the key locations of the environment and whose uncertainty is not explicitly maintained. As a result, $\mathbf{x}_k$ in topological SLAM only describes the location of the robot in the graph and is therefore a 1D random variable whose range is the set of integer values associated with the nodes of the map graph. The solution to the SLAM problem is described by the posterior probability density function (pdf, or probability mass function – pmf – to be exact) $P(\mathbf{x}_k \mid \mathbf{Z}_{0:k}, \mathbf{U}_{0:k})$, incorporating all the previous observation ($\mathbf{Z}_{0:k}$) and motion ($\mathbf{U}_{0:k}$) information of the robot. $P(\mathbf{z}_k \mid \mathbf{x}_k)$ is the observation likelihood. $P(\mathbf{x}_k \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})$ is the location prior with $P(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k)$ as the motion model from $k-1$ to k (see the prediction step in Equation (1.1)). The SLAM problem as defined by (3.2) has been successfully solved with the Markov Chain Monte Carlo (MCMC) approach [114] leading to, for example, the well-known Monte Carlo localization (MCL) algorithm [28]. We adopt the MCMC approach, specifically particle filtering to solve (3.2).

In our method, a particle corresponds to a key location in the map and the particle set χ_t describes the sampled representation of $P(\mathbf{x}_k \mid \mathbf{Z}_{0:k}, \mathbf{U}_{0:k})$

$$\chi_t = \{\mathbf{x}_k^{[1]}, \mathbf{x}_k^{[2]}, \cdots, \mathbf{x}_k^{[M]}\} \tag{3.3}$$

$$\mathbf{x}_k^{[i]} \in \{1, 2, \cdots, N\} \tag{3.4}$$

where $\mathbf{x}_k^{[i]}$ is a particle at time k and M is the total number of particles used to approximate the pdf of the robot location in the map of N key locations. As the robot navigates, each newly acquired key frame [129] is either determined to be a loop closure or a new key frame. The former case leads to adding a link to the map graph and possibly modifying the representation of the node being closed. The latter case results in a new node of the map graph, and this is handled by incrementing N by 1 and recording the image associated with the node. In this way, both localization (the former case) and mapping (the latter case) are handled and in our method, the state augmentation does not change the existing posterior pdf, as was done in [2, 23].

3.3.1 Motion Model

The motion model has the form

$$\mathbf{x}_k^{[i]} \sim P(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{[i]}, \mathbf{u}_k) \tag{3.5}$$

where $\mathbf{u}_k$ is the robot control at the current time. A simple motion model is used in our method: the robot will move one step forward with a high probability given the current location [1]. In addition, we wish to account for the motion uncertainty by allowing larger forward motion with low probabilities.

3.3.2 Update and Resampling

When the measurement $\mathbf{z}_k$ is obtained, the weights of the particles w_k are calculated according to their likelihoods given $\mathbf{z}_k$:

$$w_k^{[i]} = P(\mathbf{z}_k \mid \mathbf{x}_k^{[i]}) \tag{3.6}$$

After calculating the weights of all particles, we resample them according to their weights and add noise to the particle set by randomly generating αM ($0 < \alpha < 1$) particles to maintain the diversity of the population.

Unlike the previous work, we only use Gabor-Gist in our method to calculate the likelihood and do not need to extract keypoints [1,2] or match regions [101]. In addition, we use only a single monocular vision rather than a panoramic vision [84], to reduce computational cost and memory usage. We also use PCA to enhance the discriminating ability of the original descriptors and reduce the dimensionality. The implementation used in our study can be found online[1]. Basically, we use 20 filters applied on all three channels of a RGB color image partitioned into 16 tiles. Therefore, the dimensionality of the descriptor is 960 and each dimension is the average pixel values in the tile after filtering.

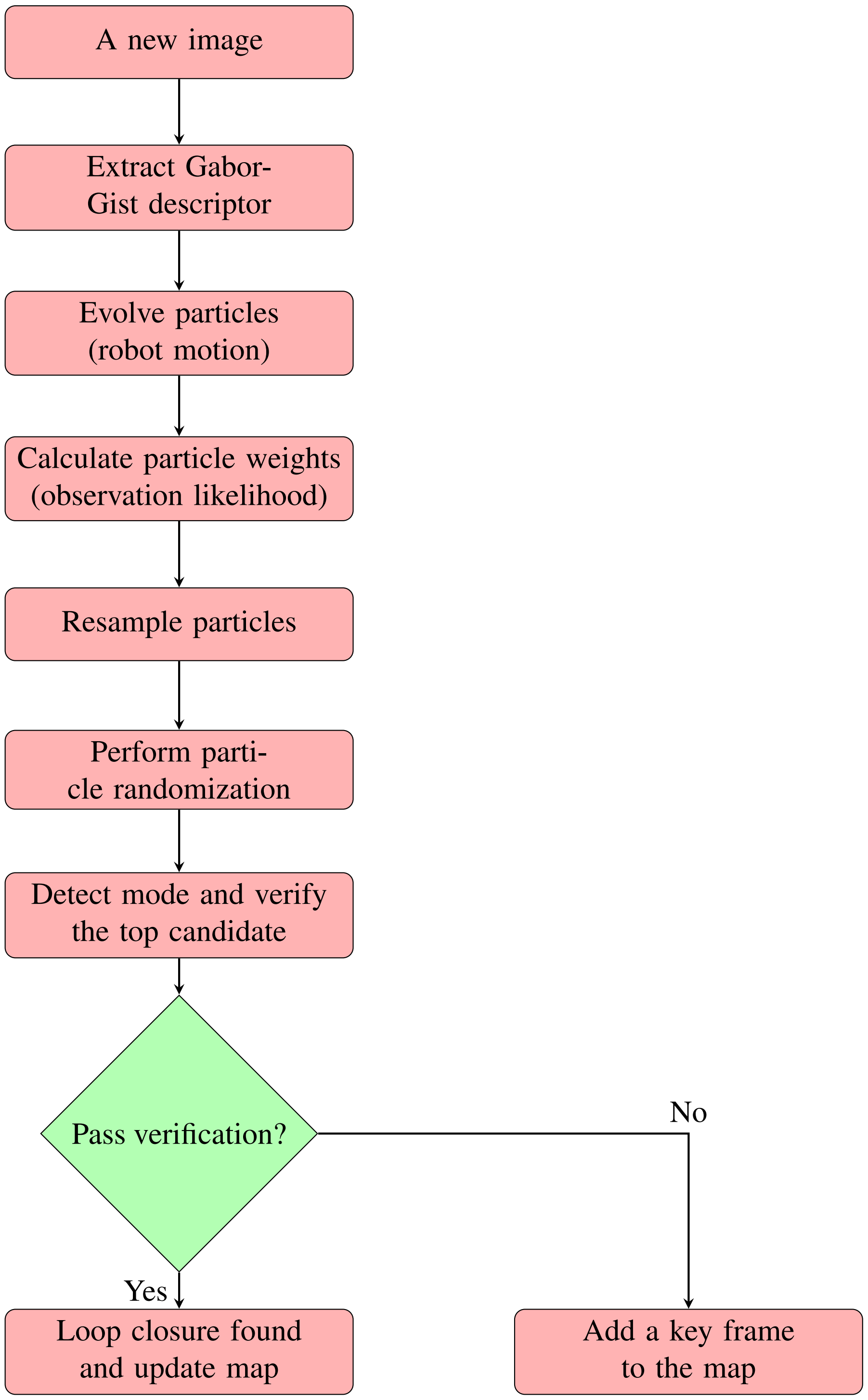
3.3.3 Monte Carlo Loop Closure Detection Algorithm

Our loop closure detection algorithm is summarized in Algorithm 3.1 using notations in [114], and a flow char is shown below. We use $\bar{\chi}_k$ to denote a temporary particle set at each step to keep the particles with their weights before resampling. In line 24, mode(χ_k) means the mode of the posterior pdf and it corresponds to the most likely loop closure candidate. In our method we estimate this mode by the location with the highest number of particles. This is just one of the several ways to estimate the mode and select the loop closure candidate, although we found it to work well in our experiments. The selected candidate is subjected to a further verification as described in Chapter 2. α is set 0.2 here.

[1]http://people.csail.mit.edu/torralba/code/spatialenvelope/

Algorithm 3.1: Loop Closure Detection Algorithm

```
$N \leftarrow 0, k \leftarrow 1$;
while $\mathbf{z}_k$ exists do
    if $k == 1$ then // initialization
        $N \leftarrow N + 1$;
        $\chi_k = \{\mathbf{x}_k^{[i]} = N \mid i = 1, 2, ..., M\}$;
    else
        $\bar{\chi}_k = \emptyset$;
        for $i = 1 \rightarrow M$ do // motion
            $\mathbf{x}_k^{[i]} \sim P(\mathbf{x}_k \mid \mathbf{u}_k, \mathbf{x}_{k-1}^{[i]})$;
            $w_k^{[i]} = P(\mathbf{z}_k \mid \mathbf{x}_k^{[i]})$;
            $\bar{\chi}_k = \bar{\chi}_k + \langle \mathbf{x}_k^{[i]}, w_k^{[i]} \rangle$;
        end
        $\chi_k = \emptyset$;
        for $i = 1 \rightarrow M$ do // resampling
            draw $j$ with the probability $\propto w_k^{[j]}$;
            $\chi_k = \chi_k + \mathbf{x}_k^{[j]}$;
        end
        for $i = 1 \rightarrow \alpha M$ do // randomization
            $u_1 \sim U[1, N]$;
            $u_2 \sim U[1, M]$;
            $\mathbf{x}_k^{[u_2]} = u_1$;
        end
    end
    $\hat{\mathbf{x}}_k = \text{mode}(\chi_k)$;
    if verify$(\hat{\mathbf{x}}_k)$ then
        loop closure detected, update map topology;
    else
        $N \leftarrow N + 1$;
    end
    $k \leftarrow k + 1$;
end
```


A new image
Extract Gabor-Gist descriptor
Evolve particles (robot motion)
Calculate particle weights (observation likelihood)
Resample particles
Perform particle randomization
Detect mode and verify the top candidate
Pass verification?
No
Yes
Loop closure found and update map
Add a key frame to the map

3.4 A Simple Extension to Sequence Matching

Our method described in Section 3.3 can be easily extended to sequence-based SLAM [73, 74, 76] which was developed to address the perceptual (especially illumination) change in long-term mapping. The main idea in SeqSLAM is to use a series of consecutive frames as the observation and matching is done between image sequences. For sequences with the same originating location, different lengths are considered so that the possible changing velocity in the robot motion can be taken into account. While the comparison between sequences can be feasible in a regular map with thousands of key locations, it may become intractable when the map size increases to, for example, millions of locations. By applying our method in sequence-based matching with a two dimensional (2D) particle filter, the run time can be reduced significantly with equivalent detection accuracy, greatly improving the efficiency in localization and mapping, and making the algorithm highly scalable in large maps.

In this extension, a 2D particle corresponds to an image sequence with a specific length $L_k^{[i]}$, originating at a key location in the map with the index $I_k^{[i]}$

$$I_k^{[i]} \in \{1, 2, \cdots, N - R_{recent}\} \tag{3.7}$$

$$L_k^{[i]} \in \{L_{min}, L_{min}+1, \cdots, L_{mid}, \cdots L_{max}\} \tag{3.8}$$

where R_{recent} is used to exclude sequences that are too close to the current robot view [76]. $L_k^{[i]}$ is in the range of L_{min} to L_{max}. We use L_{mid} as the length of the current observed sequence, meaning that each candidate sequence can be either shorter or longer than the current observation. Therefore, $\mathbf{x}_k^{[i]}$ can be one of the combinations of $(I_k^{[i]}, L_k^{[i]})$, meaning that the sequence originates from $I_k^{[i]}$ and has the length of $L_k^{[i]}$.

$$\mathbf{x}_k^{[i]} \in \{(I_k^{[i]}, L_k^{[i]})\} \tag{3.9}$$

As the robot navigates, the current sequence observed (including the current view and its predecessors within L_{mid}) is determined to be either a match to a previous sequence or a new one. In the former case the current view I_c is considered an old location while the latter case judges I_c as a new place. The map will then be modified accordingly by either adding a link to a previous node or creating a new node and incrementing N by 1.

3.4.1 Observation Likelihood

The evaluation of observation likelihood of $P(\mathbf{z}_k \mid \mathbf{x}_k^{[i]})$ now becomes complicated because $\mathbf{x}_k^{[i]}$ involves a set of locations as in Equation (3.9). A reasonable and intuitive way as in [76] is to correspond each location in $\mathbf{x}_k^{[i]}$ to its closest counterpart in the candidate sequence and this can be done via interpolation after the two sequences are aligned with each other. This method implies that the spatial information in terms of the sampling order of the frames reflects the true correlation among these frames. A simpler alternative is to ignore this correlation and assume the independence of locations in $\mathbf{x}_k^{[i]}$. Then we can relate each location in $\mathbf{x}_k^{[i]}$ to its best matching counterpart in the candidate sequence, in terms of similarity, rather than spatial arrangement. We refer this method as *maximum* estimation because it maximizes the similarity between the current sequence and a candidate one. Both estimation methods are tested in our experiments. The results do not show any difference in performance when precision is high, although at a lower precision, the maximum based estimation slightly outperforms the interpolation in [76], as will be shown in the Section 3.5.

3.4.2 Motion Model and Resampling

In the 2D filter, $I_k^{[i]}$ describes the robot motion and $L_k^{[i]}$ gives the change of the sequence length. The motion of $I_k^{[i]}$ is similar to that in Sectoin 3.3.1. The second dimension of sequence length can be unchanged, longer or shorter with equal probability. In this way we can sample the 2D space with the particles. The resampling is also similar to that in Section 3.3.2.

3.4.3 2D Particle Filter Algorithm

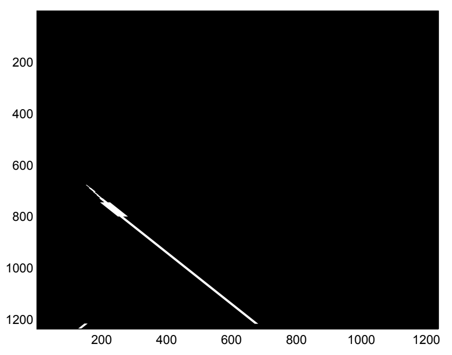
The algorithm runs a standard particle filtering process as in Algorithm 3.1 with each of the components elaborated above. There is one obvious difference in this 2D case compared with the 1D particle filter in finding the best candidate from the pdf mode. Directly selecting the hypothesis with the most number of supporting particles can be unreliable in a 2D space, especially when only a small number of particles are used. We have observed in our experiment that the originating location of a sequence is more important than the length in finding a correct match. Therefore, an intuitive way is to marginalize the second dimension to obtain a 1D distribution of the starting locations in estimating the true match and for the best one, we return all the sequences with different lengths (the second dimension). The best matching sequence is the one with the highest support by the particles. In general, the first step of marginalization strengthens the confidence in selecting the correct starting location and the second step generates the best candidate based on this location. This is of course a simple way of determining the match, although it works well in our case. More sophisticated methods can be exploited here to possibly obtain better performance. When the best matching sequence is selected, the image in this sequence with the highest similarity above some given threshold to the current robot view is deemed to be a loop closing location, otherwise the current view is considered a new place. Strict geometrical verification can be also applied in the last step to exclude any false positive.

3.5 Experimental Results

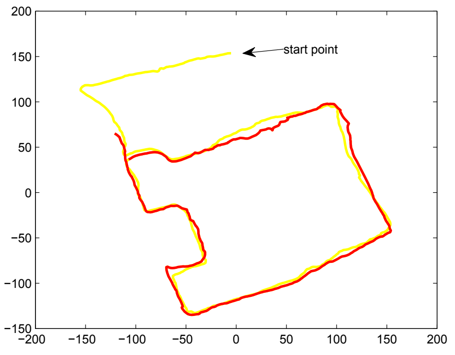
Comprehensive experiments were conducted for both methods. For the ordinary particle filter algorithm proposed in Section 3.3, we ran the experiments on two outdoor datasets with different levels of illumination change. A much larger dataset consisting of more than 12000 stree view panoramas was used in testing the performance and efficiency of the method of 2D extension.

3.5.1 Particle Filter in Appearance SLAM

The first dataset used in the evaluation is the Oxford City dataset that was published for the evaluation of FAB-MAP [23]. The dataset contains 1237 pairs of images, taken by the left and right cameras on the robot as it was driven through the environment. GPS information and the ground truth were provided. Figure 3.1 shows the ground truth matrix of the dataset and the visualization of GPS positions. More



(a)



(b)

Figure 3.1: (a) Ground truth of the dataset. Bright indicates a match between two locations. (b) Visualization of GPS positions. Note that the coordinates are only for visualization purpose and do not reflect any metric information of the map. There are two loops. The second loop is in red.

information about the dataset can be found online[2].

Motion Model

Our motion model is described as follows

$$\begin{cases} P(\mathbf{x}_k^{[i]} = n+1 \mid \mathbf{x}_{k-1}^{[i]} = n, \mathbf{u}_k) = p1 \\ P(\mathbf{x}_k^{[i]} = n \mid \mathbf{x}_{k-1}^{[i]} = n, \mathbf{u}_k) = p2 \\ P(\mathbf{x}_k^{[i]} = n+2 \mid \mathbf{x}_{k-1}^{[i]} = n, \mathbf{u}_k) = p3 \\ P(\mathbf{x}_k^{[i]} = n+3 \mid \mathbf{x}_{k-1}^{[i]} = n, \mathbf{u}_k) = p4 \end{cases} \tag{3.10}$$

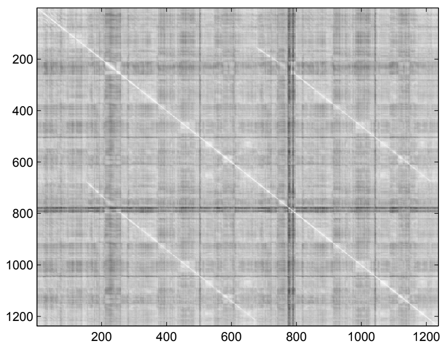
$$n \in \{1, 2, \cdots, N\} \tag{3.11}$$

In our implementation $p1$, $p2$, $p3$ and $p4$ are set to be 0.7, 0.1, 0.1 and 0.1, respectively, implying that the robot moves one step forward at each time with high probability, although it can move more than one step or stay unchanged with low probability.

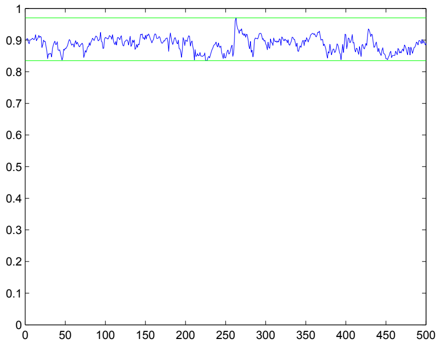
Gabor-Gist Image Descriptor

Figure 3.2a shows the result of the cosine similarity among all images using original Gabor-Gist descriptors. The left and right sets of images were processed separately and the final similarity is the mean value of the two calculations. Although the

[2]http://www.robots.ox.ac.uk/∼mobile/IJRR_2008_Dataset/data.html



(a)

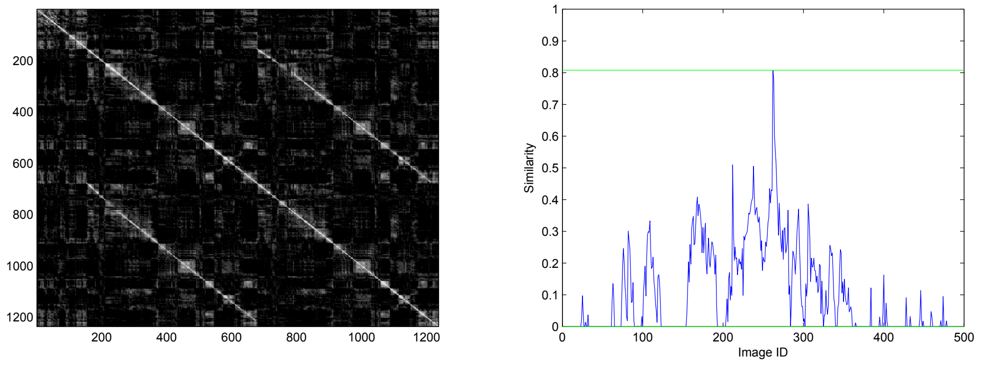


(b)

Figure 3.2: (a) Similarity matrix built with the original Gabor-Gist descriptors. The loop closure events can be visible. However, the discriminating power is limited. (b) A detailed plot of the likelihoods associated with image 800. No significant difference on likelihood exists among images.

secondary diagonal indicating loop closure is still faintly visible, the differences among all the likelihoods are not obvious. We took a random image (number 800) to plot the likelihoods associated with it shown in Figure 3.2b. The correct matches of image 800 according to the ground truth are from 257 to 267. However, it can be seen that there is no obvious difference in measurement for all the images. We have observed in our experiments this likelihood provides little help to the particle filter since it is too poor to correctly denote the temporal coherence in the image sequence, as discussed in the Section 3.3.

Nonetheless, the faint secondary diagonal corresponding to the loop closure images encourages us to optimize the result. The similarity matrix after applying PCA to the original descriptors is shown in Figure 3.3a. We reduced the original dimension of 960 to only 60, preserving 90% of the information according to the eigenvalues of the principal components. The similarity matrix is discriminative now and the loop closure is clearly visible as the secondary diagonal. To better understand the difference, we took the same image 800 to see the detail in Figure 3.3b. Apparently, the temporal coherence in the image sequence is better reflected by the gradual and obvious change on likelihood around the maximum value. Figure 3.5a further describes this change in a zoomed-in view, focusing on the neighboring images around the one with the highest value of likelihood. For the online imple-



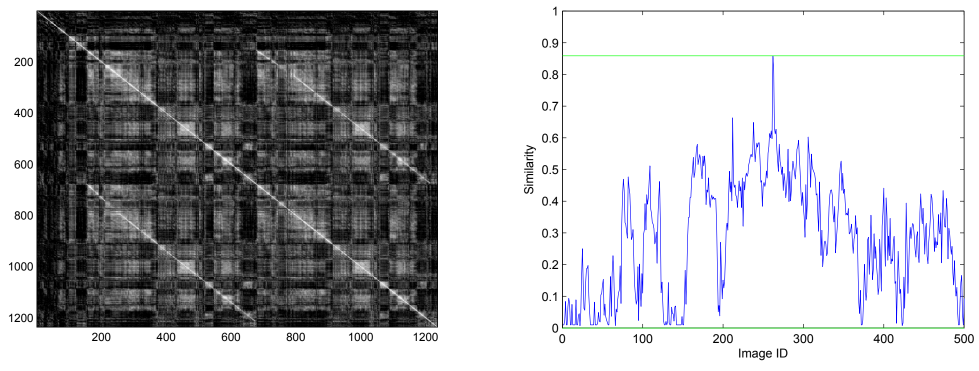
(a) (b)

Figure 3.3: (a) Similarity matrix built with the descriptors after applying PCA to the original ones. The loop closure events can be clearly visible. The dimensionality of the new descriptors is 60. (b) A detailed plot of the likelihoods associated with image 800. The difference is obvious among images.

mentation, we can sample images from the same or similar environment to obtain the principal components. Figure 3.4 and Figure 3.5b show the results of using the principal components trained from the first 100 images of the dataset.

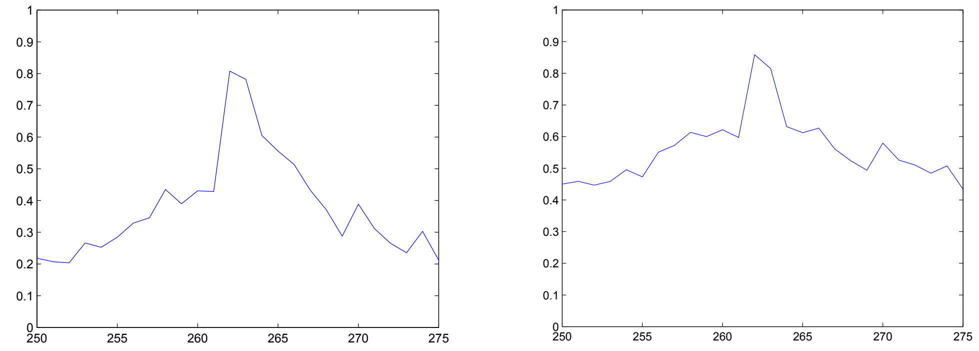
Loop Closure Detection Performance

We ran the experiments with different number of particles for repeated times to gather the performance. With the same parameter settings, the performance of all the runs (we tried up to 50 times) are almost the same. The tiny difference can be due to the random factors such as the resampling of particles. Therefore, we report the result from one of these repeated runs here for each type of parameter settings. For the verification of the loop closure candidate, we introduced three constraints to avoid excessive and unnecessary trials. 1) The location must have sufficient number of particles. The threshold here is 20% of the total. 2) The similarity measurement based on Gist descriptor with the current robot view should be higher than 0.3 and 3) the candidate should be “far enough” from the current one. These are all adjustable parameters and the purpose of using these constrains is to keep the true loop closure locations as many as possible for the further verification while excluding incorrect ones. One can of course ignore these constraints (i.e., verification is applied at every step) to obtain a higher recall, at the expense of spending more time on verification.



(a) (b)

Figure 3.4: (a) Similarity matrix built with the descriptors after applying PCA to the original ones. The principal components are trained from the first 100 images of the dataset. The loop closure events can still be clearly visible. (b) A detailed plot of the likelihoods associated with image 800. As can be seen, there is obviously better performance compared with Figure 3.2



(a) (b)

Figure 3.5: (a) A zoomed-in part of Figure 3.3b. (b) A zoomed-in part of Figure 3.4b. The likelihood of nearby locations around the one with the highest value (image 262) are shown. The smooth change reflects the temporal coherence in the image sequence.

We emphasize that the verification does not have significant impact on the run time of the algorithm since only one candidate is verified at most.

Figure 3.6 shows the precision-recall curves with respect to the verification threshold (i.e., the number of matching features between two images) with different number of particles. Generally, the performance becomes better with the increase of particles. The best recall obtained at 100% precision with 50, 100, 150 and 200 particles are respectively 0.83, 0.87, 0.85 and 0.87. It seems that 100 particles provides slightly better performance than 150 particles here. We interpret this as the convergence of the performance with the increase of particles.

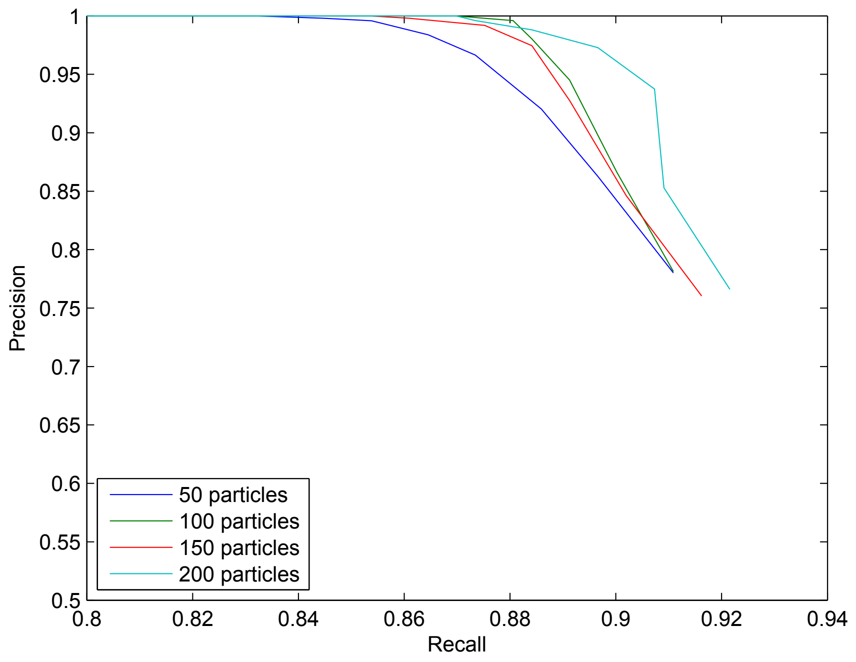


Figure 3.6: Precision-recall curves with different number of particles. Generally, the performance goes up with the increase of particles

The visualization of loop closure detection at 100% precision is shown in Figure 3.7. Note that 100% precision is achieved by using a strict verification step such as multi-view geometry as in the previous work [1,2]. We can see that most locations in the second loop are verified. However, the verification may reject some true positives. An example is given in Fig 3.8. Our result shows that many rejected true hypotheses do not share a common view with the current one. The algorithm also fails to detect loop closures for the last few locations. The reason is that the robot traverses in an opposite direction. Therefore, the motion and observation are both inconsistent with our assumptions. An example of a positive loop closure detection is given in Figure 3.9. Figure 3.9a shows the likelihoods (upper) of all the hypotheses and the corresponding pdf (lower) of the robot location which is consistent to the likelihood. Figure 3.9b and Figure 3.9c are respectively the current robot view and the retrieved location, taken from the left and right cameras.

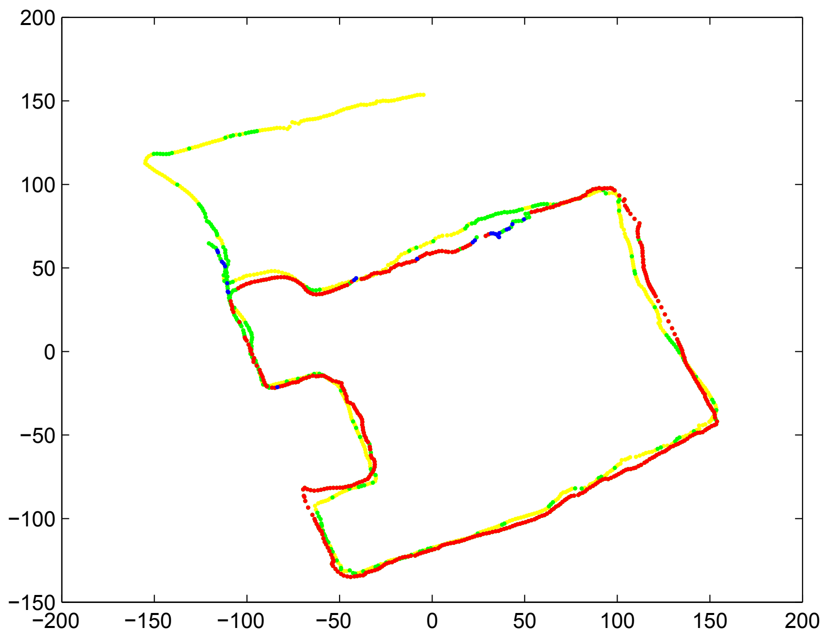


Figure 3.7: Visualization of loop closure detection at 100% precision with 100 particles. Red shows the detected loop closures (true positives). Green indicates the locations that do not pass the verification (false negatives if in the second loop). Blue illustrates the missing locations in the second loop that are not verified (also false negatives).

Processing Time

For online implementation, the processing time mainly comes from three aspects: the time of extracting the Gabor-Gist descriptor, the time of running a particle filter and the time of verification. As discussed before, feature extraction is always an inevitable process in every algorithm. Compared with local invariant features such

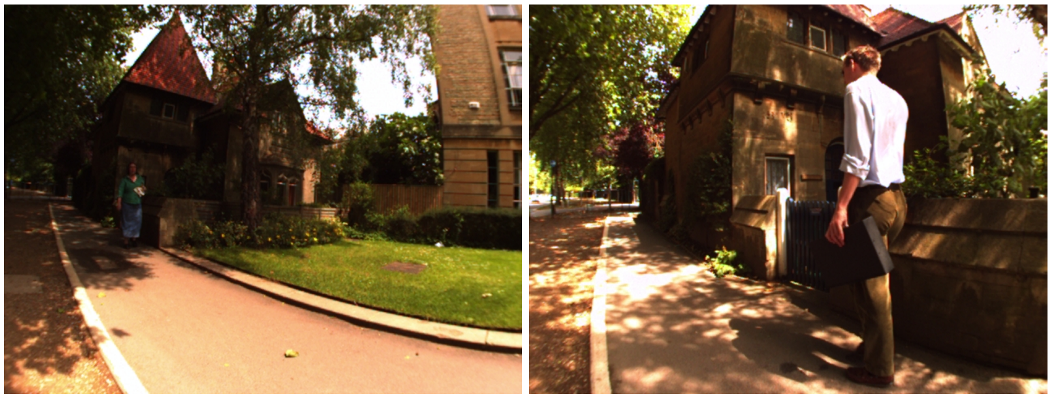
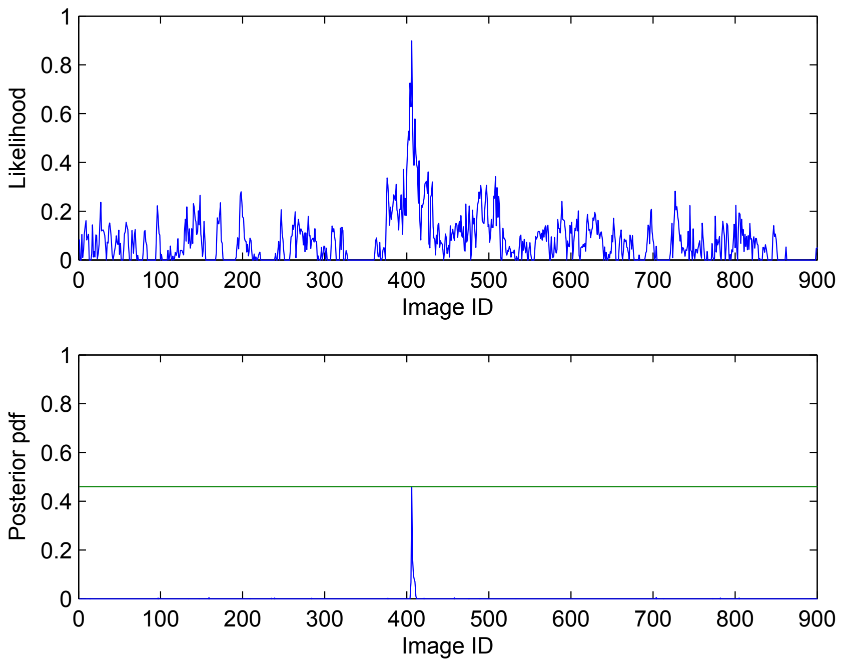
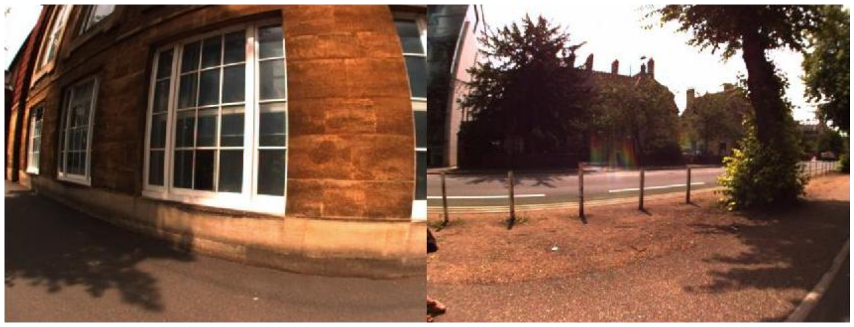

(a) The current robot view. (b) The retrieved candidate.

Figure 3.8: An example of non-verified true positive (left view of the robot). According to the ground truth, the loop closure candidate is a true match of the current view. However, the verification declines the hypothesis. It is understandable since the two views do not look similar.

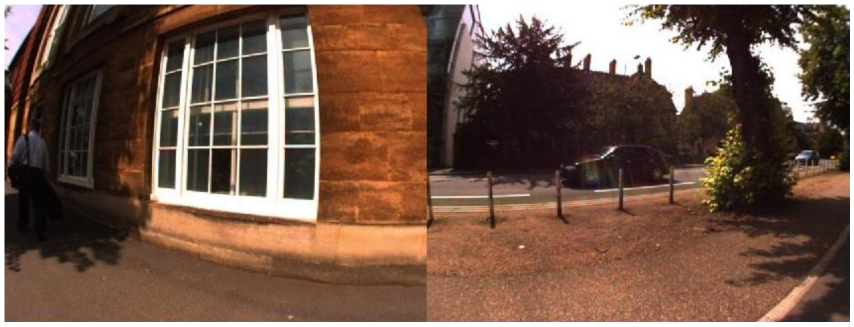
as SIFT, Gabor-Gist is faster to extract, taking only 160 ms for an image with a MATLAB implementation on a 2.40GHz lab computer. The time of running a particle filter, including the particle resampling and the motion, depends on the number of particles used in the algorithm. Our experiments show that by using 50 particles it takes only around 2 *ms* to run the filtering at each step. Even if 10000 particles are used, the implementation could still be real-time since the filtering takes up to 0.8 seconds due to the 1D nature of Equation (3.2). Regarding the likelihood calculation, the dot product used to compute the similarity measurement between the current view and a hypothesis (two 60 dimensional vectors in our case) takes negligible time. The verification can be time-consuming, depending on the number of local features extracted from the images. However, as mentioned above, this does not happen at every step, and it is a standard procedure that is used in most algorithms. Based on the above performance on loop closure detection with 50 particles in a map with 1000 locations, it is reasonable to believe that the proposed method scales easily to large maps, consisting of many thousands of locations, as will be shown later in Section 3.5.2.



(a) The pdf of the robot location and the likelihood.

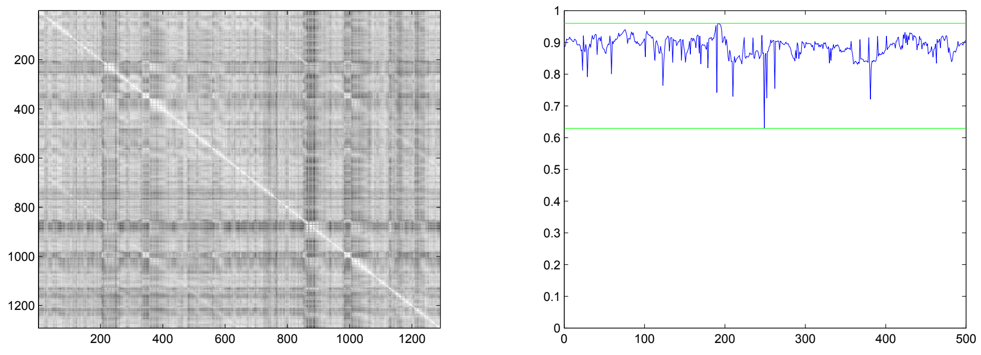


(b) The current robot view. Two images are taken from the left and right cameras.



(c) The matching loop closure image with the highest number of particles.

Figure 3.9: An example of loop closure detection. The likelihood is relatively strong around image 406. As can be seen, the likelihood and the pdf are consistent. Here 100 particles are used.



(a) (b)

Figure 3.10: (a) Similarity matrix built with the original Gabor-Gist descriptors – significant illumination change. (b) A detailed plot of the likelihoods associated with image 840.

Significant Illumination Change

Oxford City dataset conveys illumination change in a limited level. To evaluate the performance of the proposed algorithm in the case of significant illumination change, we conducted experiments on the two image sequences used in Section 2.4.2 of Chapter 2, with generally the same settings. Figure 3.10 shows the similarity matrix using original gist descriptors with an example of the detail. The results after applying PCA are shown in Figure 3.11. Apparently, as the environment becomes more challenging due to the illumination change, the loop closure events are not as distinguishable as that in Figure 3.3. However, the similarity after PCA still reflects the temporal coherence (although may not be perfect) among an image sequence. Thanks to our proposed filtering framework, we are still able to make use of the likelihoods to detect loop closures.

The evaluation of this series of experiments is slightly different from that in Section 3.5.1, mainly due to the loop closure verification. In the previous experiments, verifying a loop closure is usually not a problem since the step of establishing feature correspondences is simple when two images match. In the case of illumination change however, the true matching features are not that easy to find. As a result, even if a true loop closure is returned, it could be rejected by verification due to insufficient support from correct matches. We will discuss the problem of keypoint

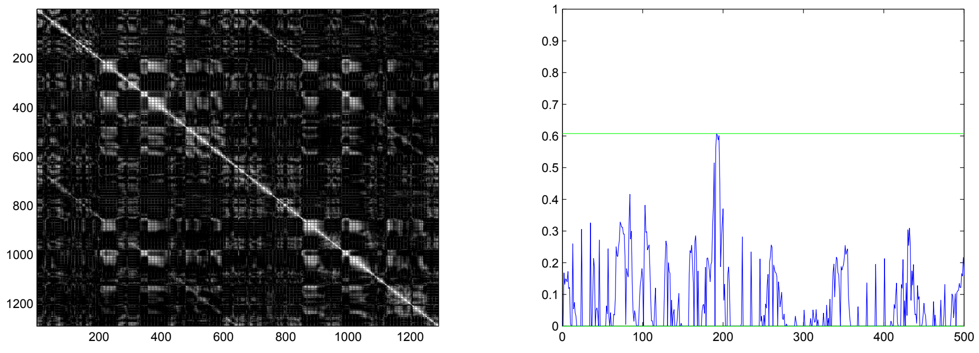


Figure 3.11: (a) Similarity matrix built with the descriptors after applying PCA to the original ones – significant illumination change (b) A detailed plot of the likelihoods associated with image 840.

matching in detail in Chapter 5 and propose an effective solution. At this moment, to fairly evaluate the proposed algorithm, we will simply assume that verification is 100% reliable and therefore consider only the recall value of detecting loop closures.

In addition, to compare our proposed method to the state-of-the-art SeqSLAM [76] which was developed to handle significant illumination change, we used its implementation on OpenSLAM[3] with default parameters. Similarly, we focused on the recall and considered the top matching for every frame in the night sequence.

Table 3.1 summarizes the recall values with respect to the number of particles, together with that from SeqSLAM. By verifying only the top one candidate, a recall value of 73% can be obtained using 500 particles. This result is better than that in Table 2.5 in Section 2.4.2 using direct feature matching, confirming the effectiveness of a whole-image descriptor plus a filtering framework. In addition, the run time is extremely fast as illustrated before, even when 500 particles are used. On the contrary, the execution of feature matching on this dataset is well beyond real-time, making it not applicable in such an environment. SeqSLAM achieves 47% recall which is much lower than the proposed method. We interpret the difference from two aspects: the gist image descriptor in this environment is more discriminating than the one used in SeqSLAM and, the filtering (not exploited in

[3]https://openslam.org/

Table 3.1: Recall w.r.t. the number of particles.

# of particles	50	100	200	300	400	500	600	SeqSLAM
Recall (%)	59.91	60.22	63.93	66.25	69.04	73.07	69.97	47.37

SeqSLAM) boosts the performance. One can definitely use the same descriptor in SeqSLAM to possibly improve the performance. That would however, result in a more time-consuming process in calculating the similarity and finding the matches, making the real-time robot exploration difficult.

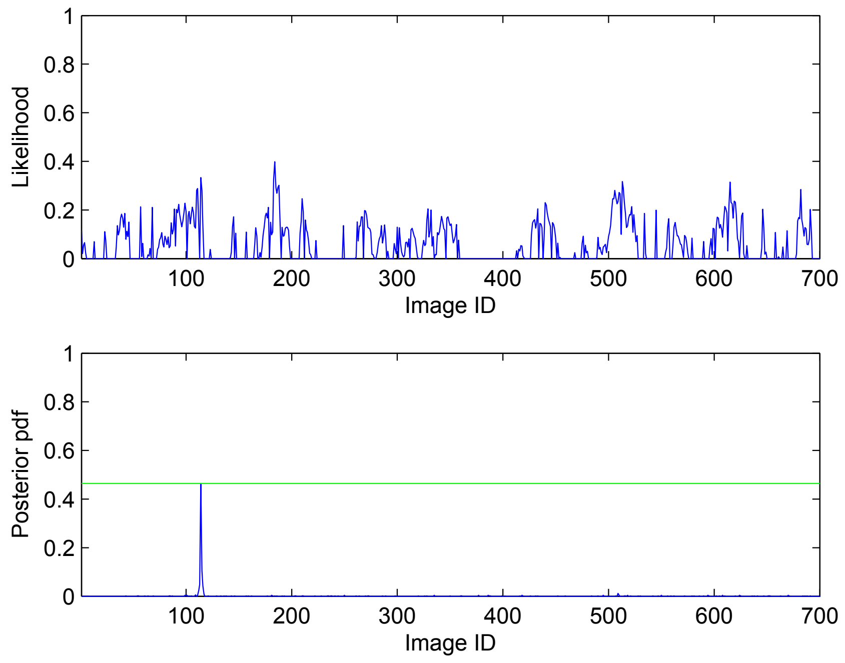
An example of loop closure detection is shown in Figure 3.12. The true matching location does not have the highest observation likelihood as shown in the top of Figure 3.12a. However, in the posterior pdf, the proposed filtering framework is able to correctly estimate the robot location due to its capability of capturing the temporal coherence in the image sequence.

3.5.2 Extension of 2D Particle Filter

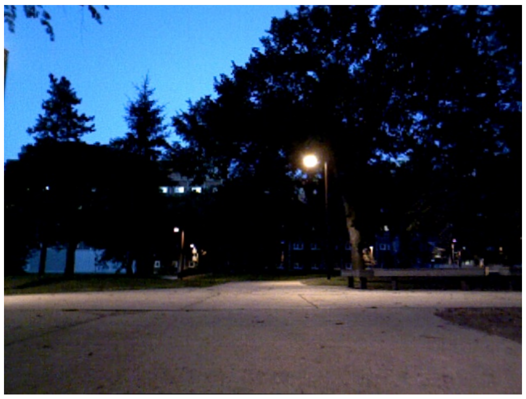
Our method has shown satisfactory results in dealing with illumination change in appearance SLAM. An important goal of evaluating the performance of the method in a 2D case is to validate its scalability. Therefore, we conducted experiments on a large dataset consisting of more than 12000 street view panoramas that are taken in the downtown area of Pittsburgh. The dataset was provided for research purposes by Google and has been used in related work [84]. The vehicle route covers a long run of 13 miles with frequent loop closure events in the process. Figure 3.13 shows in the Google map how the vehicle traverses. To make it clearer, we visualized the path in MATLAB as shown in Figure 3.14, without showing the last part of the route. Some image samples are given in Figure 3.16 and Figure 3.17.

Ground Truth

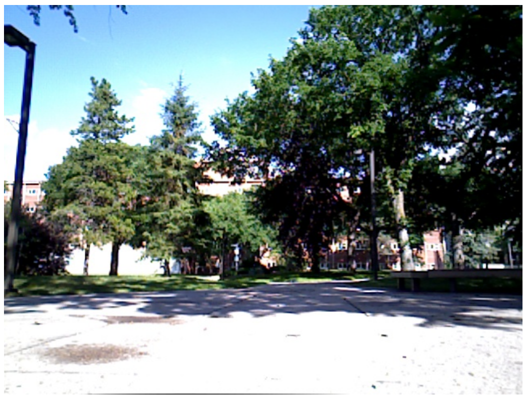
We built the ground truth for matching images ourselves as it is not provided with the dataset. For each view, GPS information (provided with the dataset) was used to select the locations that are possible revisited places. However, taking into account



(a) The pdf of the robot location and the likelihood.



(b) The current robot view.



(c) The matching loop closure image.

Figure 3.12: An example of loop closure detection – significant illumination change. The likelihood does not capture the matching location (image 114). However, the filtering is able to establish the correct posterior estimation of the robot location. Here 500 particles are used.

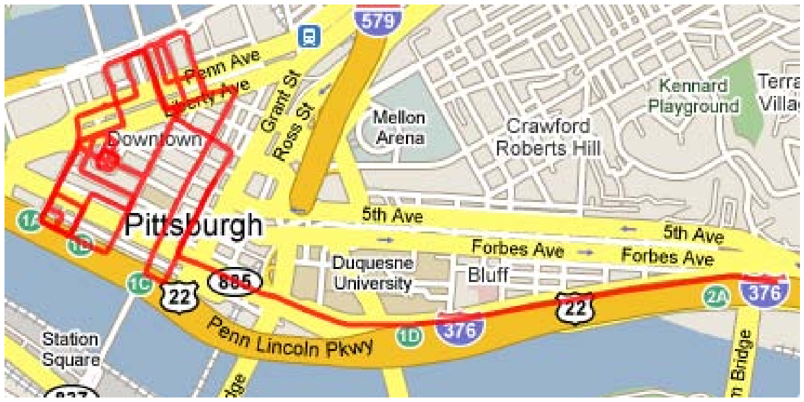


Figure 3.13: Vehicle route on Google map for the Google street view dataset. Figure is taken from [84].

the unreliable measurement in GPS, we also applied a refining process with strict visual check to finally determine the loop closure events. The guideline was that for each possible revisited place, the "best match" one with sufficient similarity to the view being checked should fall within a short distance around this view based on GPS information, and the global consistency of detection should be maintained. Here the "best match" can be determined by either visual check or any reasonable similarity measurement. The global consistency means that the detection of loop closures should be continuous without sudden interruption in the middle. This is a tedious process but will give us precise locations of loop closure events. After these steps, we produced 2941 ground truth locations with corresponding "best matches". Any location that falls within a short range of the "best match" in terms of sampling (e.g. 20 frames) would be considered a true positive.

Gist Image Descriptor

There are four sub-views for each location from the front, rear, left and right, each of which is a high resolution image of 640 $\times$ 905 pixels. After applying PCA to the original Gabor-Gist descriptors of all the sub-views, we calculated the similarity between two locations simply as the average value of the similarities from the four sub-views, which was also the method used in [109].

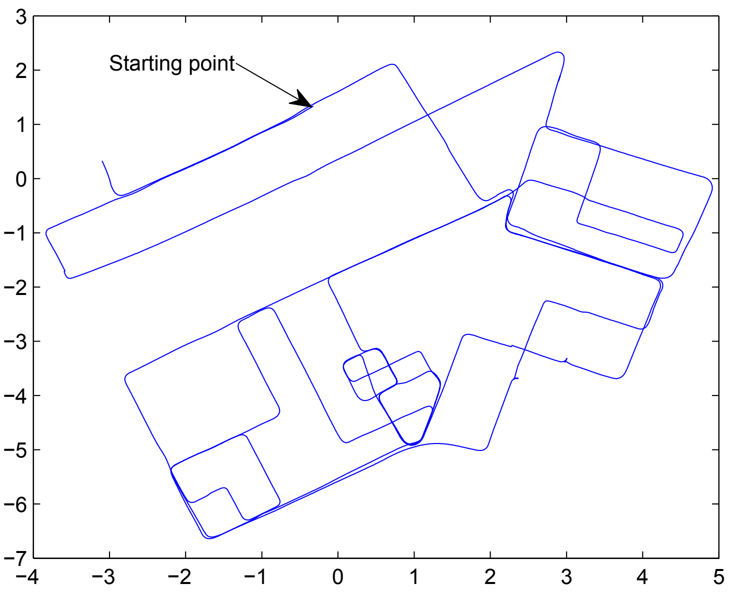


Figure 3.14: A clear visualization of the route. The coordinates in this figure does not reflect the true metric information. The last part of the route is removed for a better view.

Motion Model and Parameters

Since a 2D particle filter is used, the motion model should be applied to each dimension as described below

$$\begin{cases} P(I_k^{[i]} = n+1 \mid I_{k-1}^{[i]} = n, \mathbf{u}_k) = p1 \\ P(I_k^{[i]} = n \mid I_{k-1}^{[i]} = n, \mathbf{u}_k) = p2 \\ P(I_k^{[i]} = n+2 \mid I_{k-1}^{[i]} = n, \mathbf{u}_k) = p3 \\ P(I_k^{[i]} = n+3 \mid I_{k-1}^{[i]} = n, \mathbf{u}_k) = p4 \end{cases} \tag{3.12}$$

$$n \in \{1, 2, \cdots, N - R_{recent}\} \tag{3.13}$$

$$\begin{cases} P(L_k^{[i]} = l \mid L_{k-1}^{[i]} = l, \mathbf{u}_k) = 1/3 \\ P(L_k^{[i]} = l-1 \mid L_{k-1}^{[i]} = l, \mathbf{u}_k) = 1/3 \\ P(L_k^{[i]} = l+1 \mid L_{k-1}^{[i]} = l, \mathbf{u}_k) = 1/3 \end{cases} \tag{3.14}$$

$$l \in \{L_{min}, L_{min}+1, \cdots, L_{max}\} \tag{3.15}$$

In our implementation $p1$, $p2$, $p3$ and $p4$ are set to be 0.7, 0.1, 0.1 and 0.1, respectively. For the other parameters, $L_{min} = 15$, $L_{mid} = 20$, $L_{max} = 25$, $R_{recent} = 100$, $\alpha = 0.2$. As we have observed the performance of the algorithm is not sensitive to these parameter settings and the motion model, and currently we do not focus on the optimization of the parameters. Our primary concern is to reduce the time while still keeping a satisfactory result in detection.

Performance

We used different numbers of particles in our experiments to validate the algorithm. As a comparison, we also ran the exhaustive search method of the 2D space to find the best match for the current robot view. That is, for the current sequence, every possible previous sequence was compared and the best one was selected for place recognition. Apparently the search is fast in a small map but will become slower as the robot navigates. This will be discussed later. For the exhaustive search, we used both methods of maximum estimation in Section 3.4.1 and the interpolation-based one in [76] to calculate the similarity.

Performance in terms of precision-recall curve is given in Figure 3.15. The curves are generated by varying the threshold of the similarity measure in determining a recognition. For the two ways of calculating similarity, the results are almost identical with the maximum-based one slightly outperforms the other at a low precision. Therefore, it is not really important of choosing which one to use in our case. In the proposed algorithm, it is in general that more particles will provide better result, but the advantage will become smaller with the particle increasing, suggesting a convergence in performance as in Section 3.5.1. An interesting phenomenon is that by using 1000 particles (only 10% of the map size), our method is able to give equivalent recall (52.13%) to the exhaustive search (52.64%) at 100% precision. We do observe that the method has inferior performance at a lower precision (around 5% lower in recall). This is however, not important since in appearance SLAM, a high precision is usually required. Similar to the exhaustive search, most of the false matching candidates generated in the method can be easily excluded by using a high threshold in the decision step.

Figure 3.16 and Figure 3.17 are two examples of the perceptual aliasing in the detection. The two views in Figure 3.16 are taken from different locations but share a high similarity measure above 0.8, while the case in Figure 3.17 shows two images taken in the same spot but the similarity is below 0.5, possibly due to different

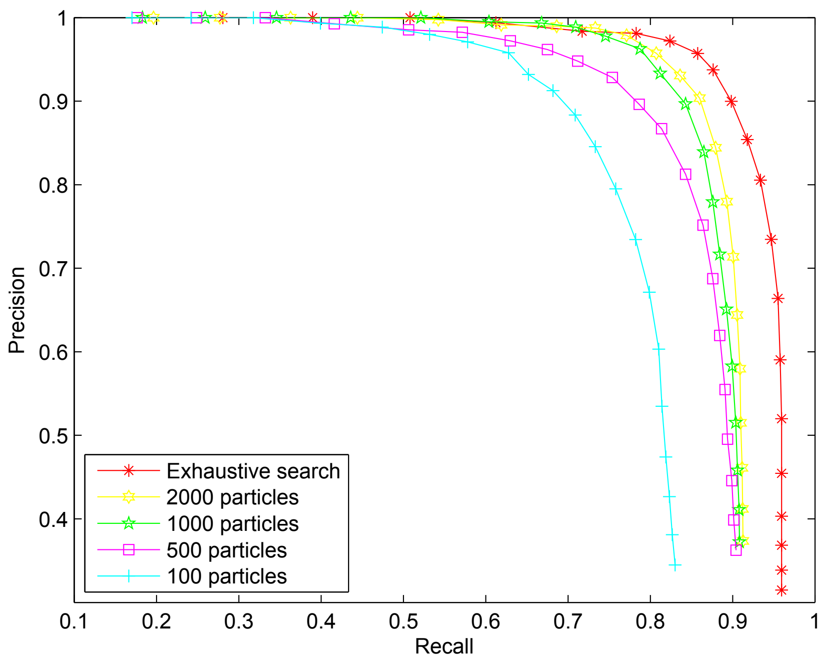


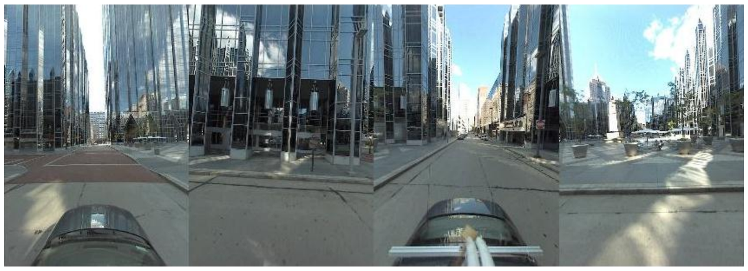
Figure 3.15: Precision-recall curves of the proposed algorithm with different numbers of particles and exhaustive search with two ways of similarity calculation. In general more particles provide better result. The proposed method performs equally well to the exhaustive search at a high precision, showing its effectiveness in this context.

illumination conditions and occlusion. These cases are not easy to overcome in both the proposed method and exhaustive search.

An example of a positive detection with 200 particles is given in Figure 3.18. Figure 3.18a shows the pdf of the marginal distribution of the first dimension, i.e., the originating location of the sequence. The one with the highest value is used to determine the best matching sequence as discussed in Section 3.4.3. Figure 3.18b and Figure 3.18c are respectively the current robot view and the retrieved true matching location.

Processing Time

The main advantage of the 2D particle filter is its efficiency in large maps. The most time-consuming part in the algorithm comes from the resampling of the particles, which grows linearly with respect to the particle number. All the other parts


(a) The current robot view of the four directions.

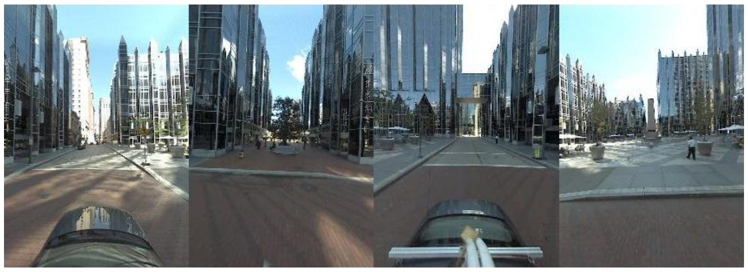
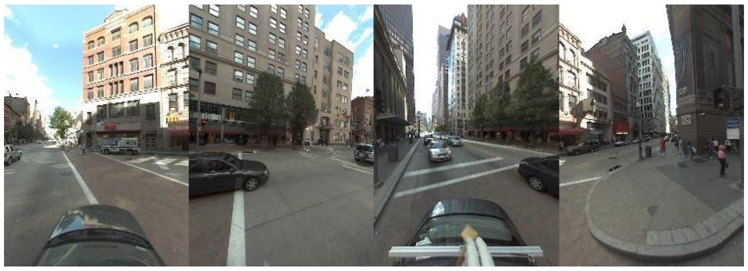

(b) The false positive retrieved as a recognition. It has a high similarity with the above panorama.

Figure 3.16: An example of the perceptual aliasing. The two panoramas are taken from different locations but share a high similarity above 0.8.


(a) The current robot view of the four directions.

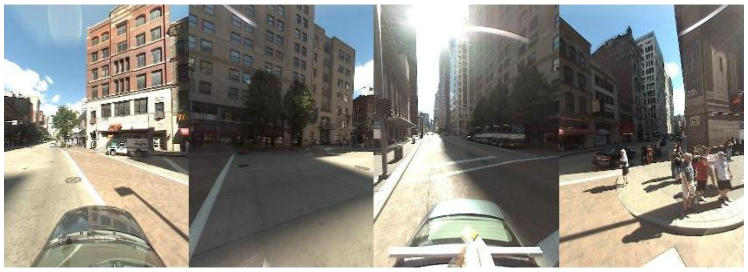
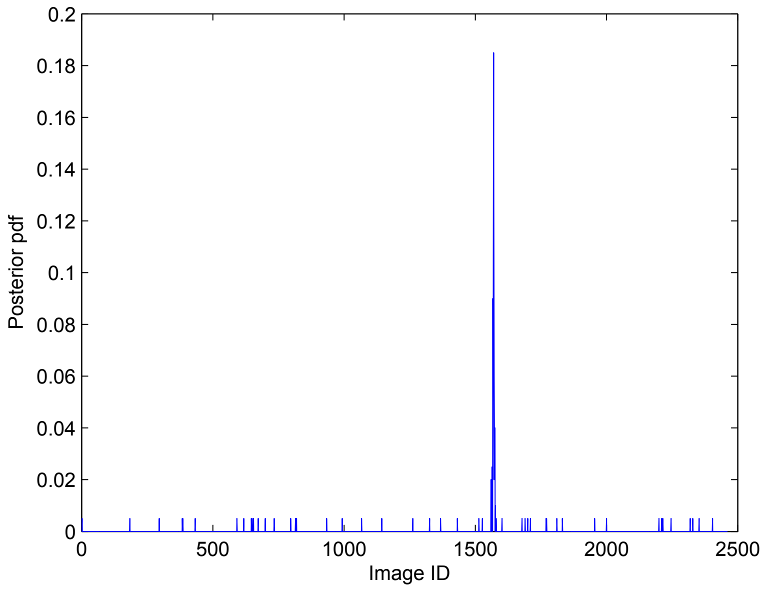
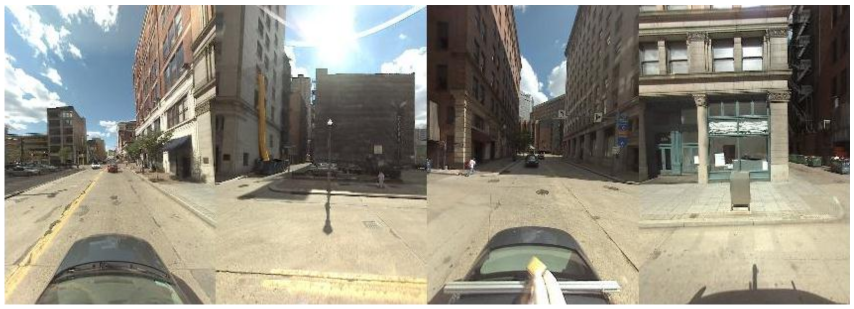

(b) The false negative missed in detection. It has a low similarity with the above panorama.

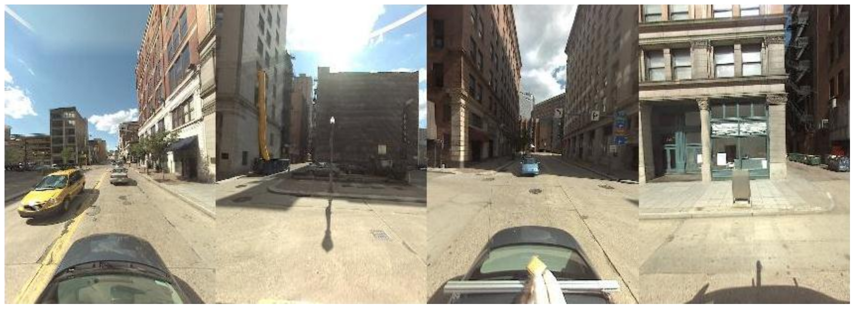
Figure 3.17: Another example of the perceptual aliasing. The two panoramas are taken at the same location but share a low similarity below 0.5, due to the illumination change (the front and rear sub-views) and occlusion (the side sub-views).



(a) The pdf of the originating location. The true matching candidates have significant higher values than the rest.



(b) The current robot view.



(c) The retrieved matching location.

Figure 3.18: An example of true detection. 200 particles are used here. The pdf gives the originating location that needs to be used for selecting the matching sequence.

Table 3.2: Speedup factor over exhaustive search at 99% precision

# particles	100	200	1000	1500	2000
Recall	47.43%	54.51%	70.93%	71.51%	73.31%
Speedup factor	139.53	78.95	15.63	9.76	6.96

such as the motion take negligible time. This resampling is independent of the map size. Therefore, the processing time is always fixed as long as the particle number is given. The time of exhaustive search however, highly depends on the map size. In the Google dataset the map size is on a 10K order and for each starting location we used 11 sequence lengths. The searching space is therefore on a l00K order. With the same implementation platform, the processing time is at least one order of magnitude longer than the proposed method in achieving similar performance. Since 100% precision is not always available due to the perceptual aliasing, we considered the recall values at 99% precision, which is 73.51% for the exhaustive search. Table 3.2 shows that even if this highest recall is desired, by using 2000 particles the proposed method can be still 7 times faster than the exhaustive search. The advantage is more obvious if a lower recall is desired. It is reasonable to believe the method can be scalable in a much larger map with satisfactory result in detection. In fact, the implementation is still in real-time even if 10K particles are used.

3.6 Summary

In this chapter, we have presented a visual loop closure detection method that uses (but is not limited to) Gabor-Gist as the image descriptor to obtain the likelihood. A particle filter framework is applied to exploit the temporal correlation in the image sequence. The method proposed in this paper is not the first one that incorporates a probabilistic framework in visual navigation. However, most previous work used the BoW image descriptor and an inverted index to compute the likelihood, and was therefore unable to exploit the MCMC approach, while in our method we use a compact global image descriptor to directly compute the likelihood for only the samples of the robot location. We have shown in our method that as long as this likelihood is able to capture the spatial correlation among neighboring locations in an image sequence, it can be used in loop closure detection or appearance SLAM with a particle filter in an efficient and effective way. In addition, a whole-image descriptor is also capable of handling illumination change where local feature-based algorithms may fail due to descriptor variance and repeatability issue. Our main contribution is to provide a novel algorithm of using a good image descriptor in visual loop closure detection in illumination change. Although it is not our intention to optimize the parameter settings, we believe that the dimensionality of the descriptor can be even lower to work on the dataset that was used in our experiments to validate the method. However, with the increase of the map size, more principal components may be needed to build the descriptor.

A simple extension of the method in a 2D case has also been described. This extension can be combined with SeqSLAM to improve the efficiency in sequence matching without too much loss of performance. It makes use of a particle filter to resample a 2D space of both the starting location of a sequence and its length. In terms of obtaining the likelihood, we have shown that a maximum-based method performs comparably with or better than sequence matching with interpolation. By using a 2D particle filter, localization and mapping in large illumination changing environment becomes possible.

As whole-image descriptors work well in appearance SLAM in dealing with illumination change, a natural follow-up question is to investigate and compare their capability in robot applications. In Chapter 4, we will conduct the performance evaluation of several whole-image descriptors in various applications to generate a guideline in applying these descriptors in a particular situation. In addition, although the proposed particle filter framework plus a whole-image descriptor can generate correct loop closure candidates, as mentioned in Section 3.5.1, the last verification step may reject many true matches in the case of significant illumination change. We will analyze this problem in Chapter 5 and provide an effective solution to ensure the detection rate.

$\sim$

Chapter 4

Performance Evaluation of Whole-Image Descriptors

In this chapter, we will present the performance evaluation of different whole-image descriptors in visual loop closure detection and other related robot applications. As mentioned in the previous chapter, a whole-image descriptor does not require keypoint detection and is therefore fast to extract. In addition, it can be extremely compact to reduce storage requirement. This type of image descriptors are attracting an increasing amount of interest in appearance SLAM or robot localization. Our evaluation is in the context of the previous works that have exploited a whole-image descriptor in the application of visual loop closure detection or robot localization. Several whole-image descriptors in three different categories are compared in our study. Our experiments are conducted on several outdoor datasets and the results show that although all these descriptors can be acceptable, they can provide significantly different performance depending upon the evaluation metrics.

4.1 Introduction

As discussed before, in the recent development of appearance SLAM or robot localization algorithms, whole-image descriptors have been used to validate the performance. It is therefore interesting to compare the effectiveness of these descriptors and set up a guideline of choosing them in the context of loop closure detection or any other applications that can benefit from a compact and discriminative image descriptor. In this chapter, we conduct the performance evaluation of different whole-image descriptors with the help of the frameworks for loop closure detection or robot localization. We first categorize the descriptors into three types according to their implementation schemes, and then select one or multiple representatives in each type for comparison. Besides the evaluation in loop closure detection, we also compare them in the kidnapped robot problem or global localization to further analyze their performance. Our purpose is to select the best and most applicable whole-image descriptor in visual robot navigation.

4.2 Classification of Descriptors

This section provides an overview of existing whole-image descriptors. We classify the whole-image descriptors into three types, according to how they are derived from an image. For each type of descriptors, one or multiple representatives are selected for comparison.

4.2.1 Filter-Based Descriptors

The Gabor-Gist image descriptor used in Chapter 3 represents an image in terms of its responses to a bank of Gabor filters. The filtered image is then divided into image tiles and the final descriptor consists of the average values of the tiles. Gabor-Gist was also used in encoding panoramic images in [84] for visual loop closure detection. For a detailed description of Gist, please refer to the original work [88]. In our evaluation, we use Gabor-Gist as the representative of the filter-based descriptors.

4.2.2 Gradient-Based Descriptors

To capture the texture of an image, gradient is widely used in many image descriptors, such as the well-known SIFT and SURF. Gradient can be calculated for pixels (SIFT) or a small neighborhood of pixels (SURF). The descriptor is a histogram of the gradient directions, and the concept can be applied to a keypoint or to the whole image, as is the case in our study.

WI-SIFT

We first downsample the image to a smaller size (e.g. 128×128) and divide it into 4×4 patches as this division was shown to provide the best result [65]. For each image patch, a gradient direction histogram with 8 bins is constructed. Each gradient direction added to the histogram is weighted by its gradient magnitude. The final 128 dimensional descriptor ($4 \times 4 \times 8$) is normalized to a unit vector.

WI-SURF

In appearance SLAM, the rotation invariance of a descriptor is often simple to achieve since the camera often moves without any rotation other than that about the vertical axis to the ground. Our WI-SURF descriptor is in fact the upright version (U-SURF) that does not take into account the orientation information, although in general, one can use the standard SURF descriptor. Similar to WI-SIFT, we use a 4×4 division of the downsampled image to build the descriptor. The filter size of the Haar-wavelet window is 4 in our experiments and the descriptor is normalized.

HoG

HoG. The histograms of oriented gradients (HoG) [25] was developed as a whole-image descriptor. It is similar to the SIFT descriptor with the difference that it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy. We include HoG in our evaluation and extract the descriptor for each downsampled image with the same size as is used in WI-SIFT and WI-SURF.

The above three gradient-based whole-image descriptors are used for evaluation because this type is the most popular implementation with wide applications. It is our interest to find if there is any obvious difference among these descriptors in the context of visual loop closure detection or robot localization.

4.2.3 Binary Descriptors

More recently, the binary feature descriptors were introduced with BRIEF [12] and BRISK [57] as the two representatives. ORB [98] is a variant of BRIEF. These binary descriptors provide efficiency in both memory usage and extraction, as well as their comparable performance with the real-valued descriptors. Since BRIEF and BRISK are similar in nature, we select the former in this study as it has been used in loop closure detection in [109], which showed that the downsampled image with 7×7 tiles provides the best performance in their experiments. Hence we also use their parameter settings in our work. The image is downsampled to a size of $m \cdot s \times m \cdot s$ pixels with $m = 7$ and $s = 48$. For each tile, a 32 byte (256 bit) BRIEF descriptor is built and the length of the final BRIEF-Gist is 1568 (32×49) bytes or 12544 bits. The size of the downsampled image used here is bigger than that is used previously (128×128). Note that however, even for a larger image, the highly efficient and compact binary descriptor can still be extracted with negligible time – much faster than the filter or gradient-based descriptors for a smaller image.

4.2.4 Pyramid Bag-of-Words

The idea of pyramid BoW was proposed by Lazebnik *et al.* in [55]. The descriptor does not belong to any of the previous types since it requires vector quantization for the extracted features in an image. However, one major difference between the pyramid BoW and the traditional BoW is that one type of pyramid BoW does not use any costly keypoint detection. Instead, it uses "weak features" which are pixels whose gradient magnitude exceeds a minimum threshold. This makes the feature extraction fast. Another key improvement is that the BoW histogram is built for increasingly fine partitions of an image and the final image descriptor is the concatenation of all the weighted descriptors along the pyramid. Therefore, the spatial information of the keypoints is contained in comparing images. In addition, the vocabulary size can be small (up to 400 visual words) and the descriptor is faster to compute with respect to the other whole-image descriptors we consider. We include pyramid BoW in our study as an alternative of the traditional BoW image descriptor.

4.3 Evaluation Framework

Performance evaluation of the whole-image descriptors is conducted under the particle filter framework proposed in the previous chapter for visual loop closure detection, which is our main focus in this study. We use the recall-precision curves of the loop closure detection algorithm as the performance metric. In general, our method considers all the previous locations together as the state space. The goal is to estimate the pdf of robot location based on a sequence of observations.

As an alternative application, we also use the global localization or the so-called kidnapped robot problem as a context for comparing the whole-image descriptors. A kidnapped robot refers the case when a robot suddenly loses tracking of its correct location in the map. In such a case, it is hoped that the robot can re-localize itself soon. Recovery from a kidnap is essentially the global localization problem, which estimates the robot location in the entire state space and is a fundamental capability for a mobile robot. Therefore we want to design another experiment to test the convergence capability of these image descriptors in the kidnapped robot problem, i.e., how soon the robot can recover from a kidnap with these descriptors as the similarity measurements as in [3].

The particle filter algorithm in Chapter 3 works equally well for the global localization problem (except that no state augmentation is required), and it is therefore used to solve the kidnapped robot problem and evaluate performance. The evaluation metric in this case is the average number of iterations it takes for the robot to re-localize itself, upon being kidnapped randomly within a previously learned map.

4.4 Experimental Results

We implemented WI-SIFT, WI-SURF and BRIEF-Gist and the other (HoG[1], Gabor-Gist[2] and pyramid BoW[3]) implementations can be found online. Default parameter settings were used for the available implementations. Therefore the dimensionalities of Gabor-Gist, HoG and pyramid BoW are 960, 81 and 4200 respectively.

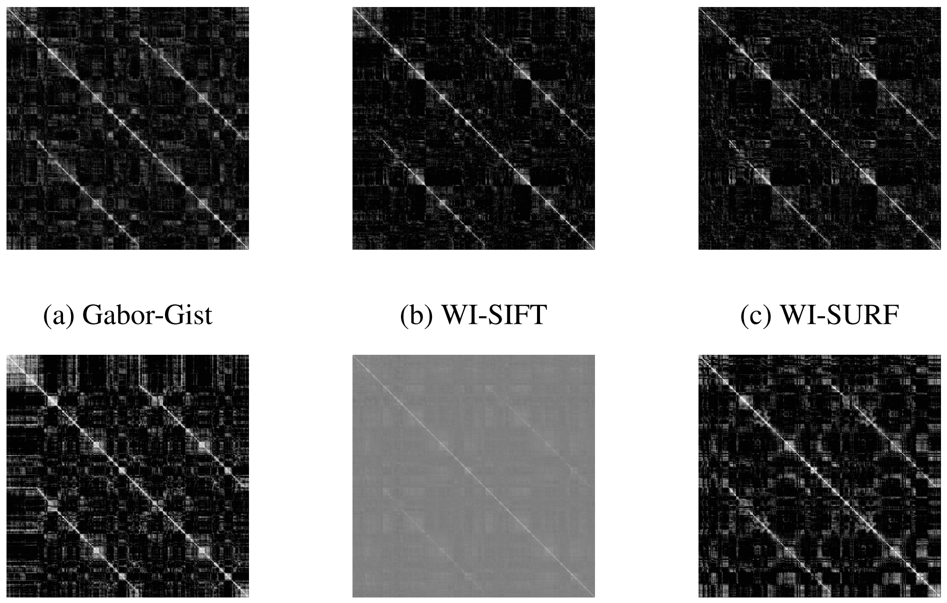
[1]http://www.mathworks.com/matlabcentral/fileexchange/28689-hog-descriptor-for-matlab
[2]http://people.csail.mit.edu/torralba/code/spatialenvelope/
[3]http://www.cs.illinois.edu/homes/slazebni/

PCA was applied on the original descriptors except BRIEF-Gist to reduce the dimensionalities of the descriptors while preserving around 90% information. This produced all the final descriptors with fewer than 100 dimensionality except for BRIEF-Gist where PCA does not apply. Two datasets of Oxford City centre and Google street view that were introduced and used in the Chapter 3 were also used in this evaluation. Note that the study conducted in this chapter mainly focuses on the performance evaluation of descriptors. We used public datasets as they have been widely used in related work in the context of robot applications. The performance of these candidate descriptors may drop if experiments are conducted in a more challenging environment, for example, with significant illumination change. However, it is reasonable to believe that the difference among the descriptors can be similar.

4.4.1 Oxford City Centre Dataset

The similarity matrices created by each whole-image descriptor after PCA (Hamming distance was used for BRIEF-Gist as the similarity measure and cosine similarity for the rest) are shown in Figure 4.1. As can be seen the loop closure events responsible for the secondary diagonal can be visible for all the cases. However in BRIEF-Gist, this secondary diagonal is not as clear as in the other cases. This suggests that the similarity among images is captured as well, although not obvious by the binary descriptor.

As a preliminary analysis, we processed the similarity matrices in Figure 4.1 and assumed the top matching image to be the loop closure event, subject to a threshold that varies from 0 to 1. This was the method adopted by [109]. Although the method does not exploit spatial coherence of the loop closure events, it is nonetheless appropriate for our performance comparison purposes. The precision-recall curves are shown in Figure 4.2. With 100% precision, Gabor-Gist provides the best recall and all the other descriptors perform similarly. At a lower precision however, WI-SIFT and WI-SURF can still give equivalent performance to Gabor-Gist, while that of the other three drops. BRIEF-Gist can be better than pyramid BoW and HoG, which suggests that binary descriptors are useful in some applications. An-


(a) Gabor-Gist (b) WI-SIFT (c) WI-SURF

(d) HoG (e) BRIEF-Gist (f) Pyramid BoW

Figure 4.1: The similarity matrices with different image descriptors for the Oxford City Centre dataset.

other interesting point is that although pyramid BoW does not show a competitive result in this evaluation, it can be useful for loop closure detection in the filtering framework when spatial coherence is exploited, as will be shown later.

Particle Filter-Based Loop Closure Detection

For each image descriptor, we repeated our experiment on the particle filter-based loop closure detection algorithm 15 times for a given number of particles. There are totally 561 loop closure events in this dataset and the recall is used as the evaluation criterion as precision can achieve 100% by using a strict MVG-based verification. Specifically, we counted the number of true loop closure hypotheses that were submitted for verification. The assumption of a reliable verification is made here, which implies that all the selected true loop closures can be confirmed by this verification and the false ones can be rejected. Therefore, the recall value here is not comparable to the recall mentioned above in Figure 4.2 with a simple thresholding scheme.

Figure 4.3 shows the evaluation for all the image descriptors except for BRIEF-Gist, which we will discuss shortly. We use the standard deviation of all the runs

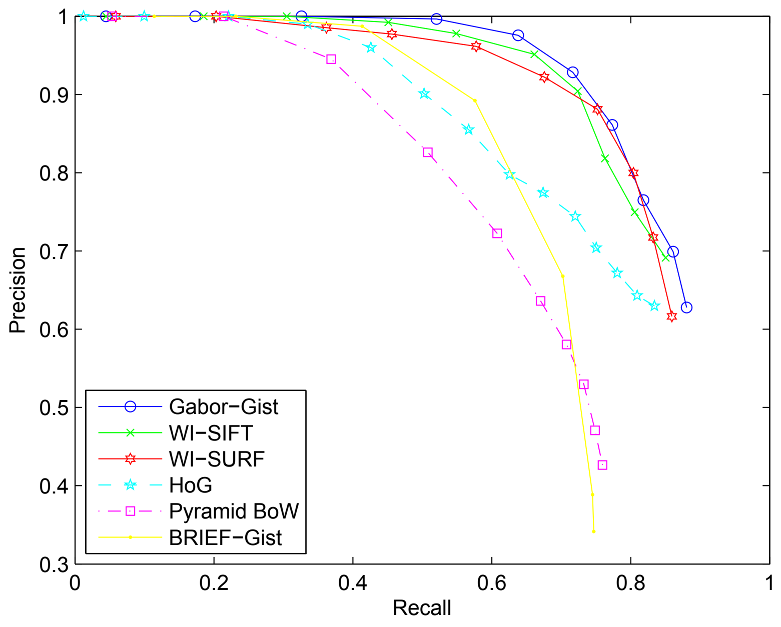


Figure 4.2: Precision-recall curves for different whole-image descriptors on Oxford City Centre dataset. The curves are generated by considering the top one match for each image and using different thresholds. At 100%, Gabor-Gist gives the best recall.

to account for the performance uncertainty. The results show that performance will become better with increasing number of particles and tend to converge. Gabor-Gist provides the best performance of all the descriptors, while pyramid BoW is the worst. This result is consistent with that in Figure 4.2. A possible explanation of the inferior performance with pyramid BoW is that the naïve weak feature is not discriminative enough. However, considering that the recall of pyramid BoW in Figure 4.2 never reaches 80%, our algorithm does help improve the performance and makes it possible to use the descriptor in loop closure detection. The three gradient-based descriptors perform almost equally, which means they have similar discriminating power under this evaluation criterion. It is also worth noting that the gradient-based descriptors provide approximately 3% lower recall than Gabor-Gist, with an obvious advantage over Gabor-Gist in computational efficiency. We will analyze the time complexity in Section 4.4.3 to compare the efficiency of the descriptors.

Our evaluation framework shows poor performance of BRIEF-Gist because of

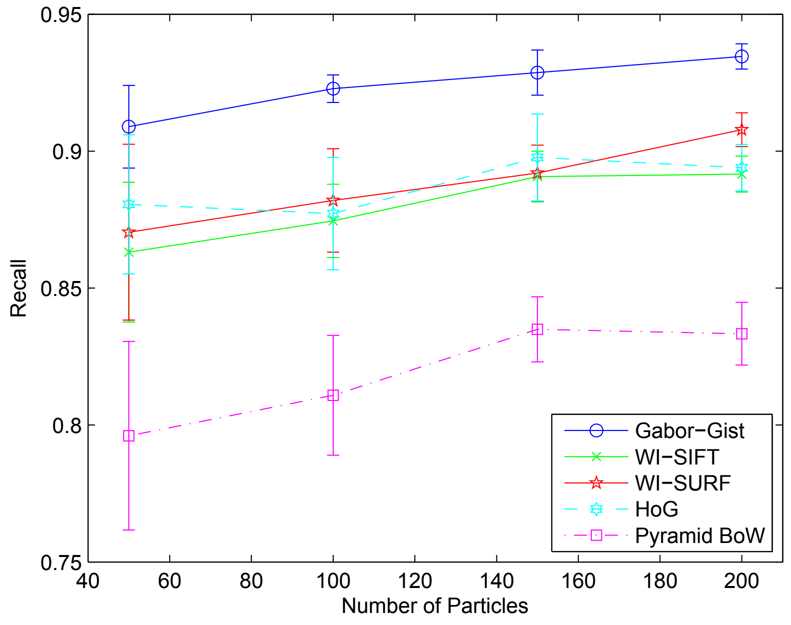


Figure 4.3: Recall of loop closure detection w.r.t. the number of particles on Oxford City Centre dataset. The recall defined here is different from it is defined in Figure 4.2. Gabor-Gist performs best. We do not include BRIEF-Gist here since it does not work with the evaluation framework.

the ambiguous similarity measurements (Figure 4.1e). The resampling based on the particle weights would not work well in such a case and since random particles are introduced in every step to maintain the population diversity, we can hardly establish a stable track for the continuously and correctly matching locations. However, this does not suggest the Bayes framework is ineffective. Similar to the PCA that can be applied on real-valued descriptors, a possible solution is to exploit methods that can improve the discriminating power of the binary descriptors, possibly an interesting open problem in binary descriptor research.

Kidnapped Robot Analysis

The whole image descriptors have also been compared in the problem of global robot localization. For kidnapped robot evaluation, the experiment was repeated 20 time with different numbers of particles and the results are shown in Figure 4.4. In this case, the performance differences among descriptors are not obvious, although Gabor-Gist still slightly outperforms the others, especially when the number of particles is increasing. Interestingly, the pyramid BoW does not show a visibly inferior result as in the previous experiment (Figure 4.2 and Figure 4.3). The possible explanation is that since the similarity measurement with pyramid BoW is capable of capturing the temporal coherence in the image sequence and therefore does help drive particles toward the correct location when a kidnap happens. However, due to its lack of discriminating power which has been shown in Figure 4.2, the tracking can be lost easily and need for relocalization arises frequently. This has been observed in our loop closure detection experiments. BRIEF-Gist (not shown) still cannot perform well due to its inability of representing the spatial or temporal coherence clearly.

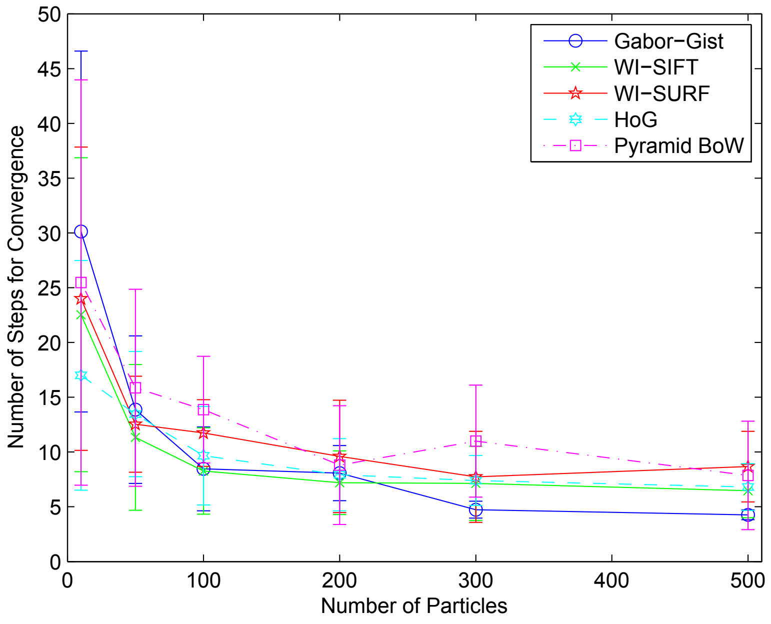


Figure 4.4: Convergence steps w.r.t. the number of particles on Oxford City Centre dataset. There are no significant differences among descriptors in this application, although Gabor-Gist still slightly outperforms the others.

4.4.2 Google Street View Dataset

Figure 4.5 shows the precision-recall curves of the descriptors produced by changing the threshold to determine if the top one match is a true positive or not. Gabor-Gist, WI-SIFT and HoG perform equally on this dataset at a high precision, followed by WI-SURF, BRIEF Gist and pyramid BoW. The results are generally consistent to that in Figure 4.2, with the difference that HoG outperforms WI-SURF. This can be understandable because each dataset is a different case and it could be true that the gradient between a neighborhood of pixels in SURF is not a proper way to capture the appearance of the images in this dataset.

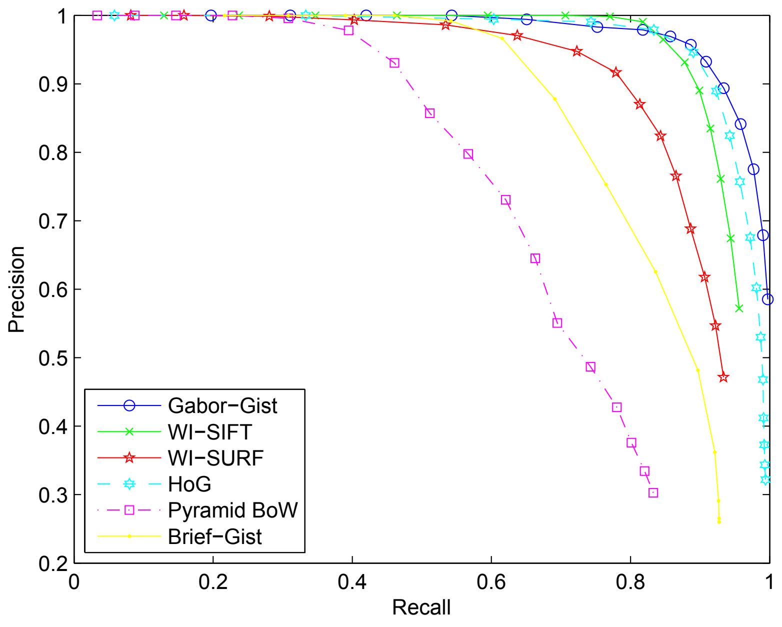


Figure 4.5: Precision-recall curves for different whole-image descriptors on Google dataset.

The performance of particle filter-based loop closure detection is shown in Figure 4.6. Since this is a large dataset, we ran once the experiments for each group of parameter settings. Figure 4.6 shows that 500 particles (about 5% of the total number of locations) can be enough to obtain converging performance for this dataset and the results are consistent to that in Figure 4.5. Compared with the previous dataset, Gabor-Gist no longer shows obviously superior performance to the gradient-based ones but it is still among the top. For the three gradient-based descriptors, WI-SIFT is the most reliable one since it provides stable performance in both datasets.

Figure 4.7 shows the performance of image descriptors in kidnapped robot problem for Google dataset. The results are generally consistent to that in Figure 4.4,

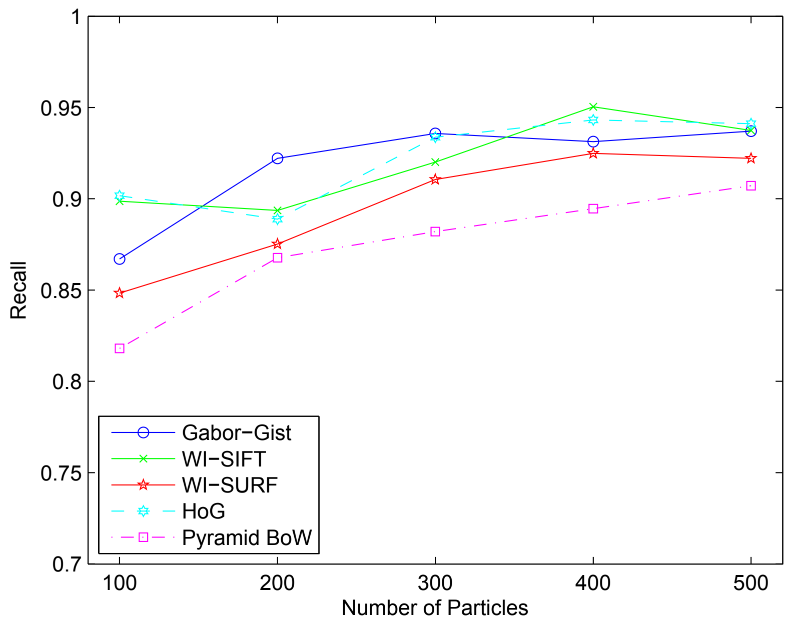


Figure 4.6: Recall of loop closure detection w.r.t. the number of particles on Google dataset.

where all the descriptors perform almost equally. In addition, there is a trend for convergence with the increase of the particle number. The standard deviation also becomes smaller with more particles, suggesting more stable performance.

4.4.3 Time Complexity

The filter-based Gabor-Gist descriptor always provides stable performance. However, it is not the most efficient one. The extraction of the filter-based descriptors usually requires a convolution and its time complexity is $O(n\text{log}n)$, where n is the number of pixels in the image. For the gradient-based descriptors, the time complexity is $O(n)$ since only one traversal over the image pixels is needed. The extraction of the binary BRIEF-Gist is in constant time regardless of the image size. The absolute execution time can be dependent on many factors such as how the implementation is optimized. In our specific case with MATLAB implementation, the WI-SIFT and WI-SURF are fives time faster to extract than Gabor-Gist for the downsampled image with the same size and BRIEF-Gist is even faster. Most importantly, even with the MATLAB implementation, all whole-image descriptors

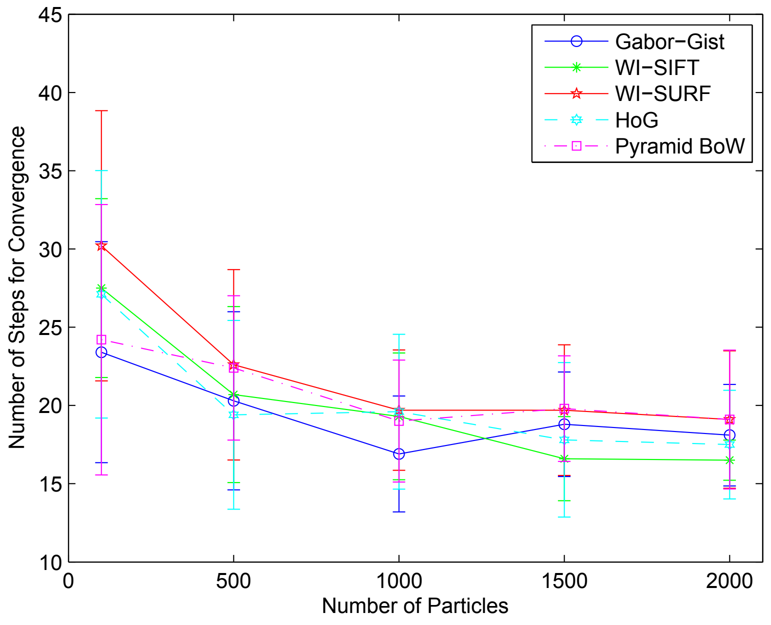


Figure 4.7: Convergence steps w.r.t. the number of particles on Google dataset. Gabor-Gist and HoG are better than the others in this application.

together with the loop closure detection algorithm work efficiently, for well under a second in processing one image, fast enough for most ground robotics applications.

4.5 Summary

In this chapter, we have presented the performance evaluation of several whole-image descriptors classified into three types, the filter-based, the gradient-based and the binary descriptors. A variant of BoW, named the pyramid BoW was also studied in our evaluation. Three evaluation metrics were used for the comparison, including the recall-precision curve of loop closure detection based on a simple analysis of the similarity matrix, the recall curve of loop closure detection using our proposed algorithm in Chapter 3 that utilizes a particle filter and the convergence rate of the kidnapped robot application also using our particle filter-based localization algorithm. Our results show that the filter-based descriptor such as the Gabor-Gist performs better than or at least equally to the others in all these applications at the expense of a higher time complexity in implementation. WI-SIFT is the most stable descriptor among the three gradient-based ones, possibly due to its strong local invariance. The binary descriptor, i.e., BRIEF-Gist, is applicable in visual loop closure detection based on the simple analysis with the similarity matrix. However, this naïve method of selecting the best match by comparing the current image with all the previous ones, may not work well in large scale applications without exploiting spatial coherence of loop closing events. In addition, a principled guideline for selecting a reasonable threshold to trigger loop closure is a difficult issue. Pyramid BoW can be acceptable in a kidnapped robot problem, but is not stable enough in a long time tracking.

Future work includes optimizing the image descriptors with respect to their tuning parameters. Taking advantage of the low time complexity, there is a potential of using BRIEF-Gist in a more effective way by exploring the methods of increasing its discriminating power.

∼

Chapter 5

Robust Keypoint Matching for Verification

In this chapter, we will propose a simple yet effective keypoint matching method that is able to perform well under significant illumination changes. Keypoint matching is an essential step in verifying loop closures. We contend and verify experimentally that a major difficulty in matching keypoints when illumination varies significantly between two images is the low inlier ratio among the putative matches. The low inlier ratio in turn causes failure in the subsequent RANSAC algorithm since the correct camera motion has as much support as many of the incorrect ones. By assuming a weak perspective camera model and planar camera motion, we derive a simple constraint on correctly matched keypoints in terms of the flow vectors between two images. We then use this constraint to prune the putative matches to boost the inlier ratio significantly thereby giving the subsequent RANSAC algorithm a chance to succeed.

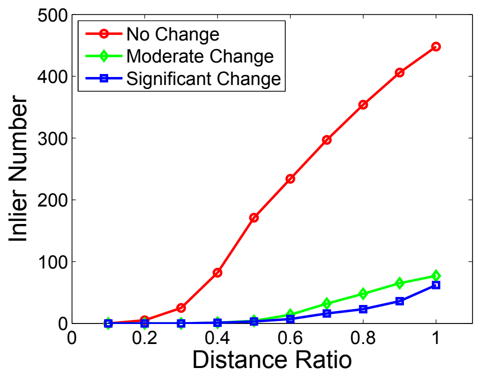
5.1 Introduction

In the previous chapters, we have discussed multiple ways of dealing with illumination change in appearance SLAM, including feature matching with *k*-d tree and the use of a whole-image descriptor in a particle filter framework. As mentioned before, the last step in appearance SLAM is the verification that identifies the true matching locations from a list of candidates generated by a loop closure detection algorithm. Verification is expected to be reliable and robust, and therefore performed in local feature level, with the expectation to find true matching keypoint pairs.

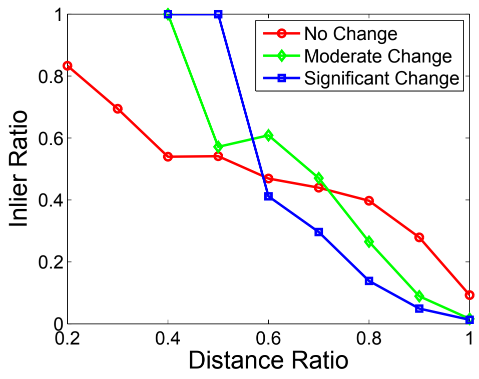
Keypoint matching is usually performed in two steps: similarity comparison where the keypoints in the first set are matched with those in the second using their descriptors, and geometric verification which imposes a constraint on the putative matches that they originate from the same camera motion. For similarity comparison, a nearest neighbor search can be used to find the putative matches, which include both the true matches (inliers) and many incorrect matches (outliers). Matching via nearest neighbor however works poorly because it generates a large number of outliers. A uniqueness constraint such as distance ratio test [65] can be applied to improve the inlier ratio in the first step. The geometric verification step focuses on finding a model of the camera motion that the true matches must satisfy using well-known results from multi-view geometry (MVG). The most popular constraint is the epipolar constraint where the matched keypoints must lie on the same epipolar lines [42]. Since neither the inliers nor the camera motion is known in general, RANSAC [33] is typically employed to search for the camera motion in the presence of outliers.

In spite of the effort in achieving invariant properties of feature detectors and descriptors, keypoint matching based solely on descriptors can be still problematic when the illumination changes significantly. Figure 5.1 gives an example why illumination change can cause considerable difficulty in keypoint matching in the standard method where distance ratio test [65] is used as the pruning step. When there is no change in illumination (red curves in Figure 5.1), sufficient true matches can be found and false matches rejected by choosing a proper distance ratio, as proposed in [65]. However, in the case of significant illumination change (blue curves in Figure 5.1), it becomes difficult to select a threshold that finds sufficient true matches without including many false matches. An attempt to increase the number of true matches by using a higher distance ratio would also unfortunately introduce a large number of outliers and cause difficulty in the subsequent RANSAC algorithm.

Our proposed method resorts to a non-parametric geometric constraint on keypoint displacements as a way of boosting the inlier ratio. Our solution is partially



(a) Inlier number w.r.t. *dr*



(b) Inlier ratio w.r.t. *dr*

Figure 5.1: Typical inlier/outlier distributions with respect to *distance ratio* under different illumination conditions. As illumination changes drastically (blue curves), it becomes difficult to find an appropriate distance ratio that can produce a high inlier ratio and a high number of inliers.

inspired by the research on the spatial statistics of optical flow [97], which finds that optical flow of images in a video sequence, captured by a hand-held camera, follows a well-defined peaked Laplacian distribution. Since the displacement vectors of matching keypoints between images is equivalent to optical flow, they might also follow a well-defined statistical distribution. In fact, this distribution can be established theoretically with the weak perspective camera model and planar camera motion, as is experienced in many robotics and computer vision applications. Thus we can employ this distribution as a prior of the inlier matches. Weak perspective is a reasonable camera model in many outdoor environments where the depth of the scene points is small relative to their distance to the camera or when many of the keypoints of interest have this property.

5.2 Related Work on Keypoint Matching

In this section, we review the background and related work of our research, including the methods of establishing keypoint correspondences and the existing solutions to this problem under illumination change.

5.2.1 Keypoint Matching with Geometric Constraints

As mentioned, keypoint matching algorithms involve two steps: (1) finding a set of putative matches and (2) removing outliers by imposing a geometric constraint on the putative matches. Keypoint descriptors are used in the first step where putative matches are obtained by measuring similarity between descriptors. In the second, a uniqueness constraint such as distance ratio and mutual consistency can be applied to exclude matches that are likely to be false. As well, one can attempt to identify an underlying camera motion between the two images and use the motion to prune putative matches. For identifying the underlying motion, there are generally two types of strategy in obtaining the inliers. One is through the iterative hypothesize-and-verify framework where a model is computed with a randomly selected subset of putative matches and verified by the rest to estimate the level of support for the model. One the other hand, the model can be also estimated using the entire set of putative matches. In such a case, keypoint matching is formulated as a combinatorial optimization problem and the inliers are associated with the optimal solution.

RANSAC [33] is an effective algorithm to identify inliers induced by the underlying camera motion in the presence of outliers. The algorithm continuously iterates until either the maximum number of steps is reached or the algorithm reaches a solution with sufficient support. Several variants of RANSAC have been developed to improve the performance of the original algorithm. For example, MLESAC [116] uses a weighted strategy to evaluate how well the inliers fit the model, taking into account the outlier distribution. A solution to maximize the likelihood, rather than the number of inliers, is selected to generate more accurate estimation of camera motion. On the other hand, LO-RANSAC [19] focuses on reducing the number of iterations in RANSAC. This can be done by adopting a local optimization step with a selected set of potential inliers to re-estimate the parameters in the current best model. Alternatively, an ordering structure of the set of putative matches based on descriptor similarity is exploited in PROSAC [18]. Samples are drawn from progressively larger sets of top-ranked correspondences and the algorithm converges much faster than RANSAC. A detailed performance evaluation and analysis of RANSAC and its variants are provided in [93]. Apparently, all the variants mentioned here are motivated by the desire to improve the model accuracy, with the potential benefit of speeding up the convergence. The issue of low inlier ratio however, is not effectively addressed in these methods, since they still depend on randomly selected samples from the entire putative matching set.

Alternatively, keypoint matching can be formulated as a constrained optimization problem where all putative matches are used as the input to an objective function. In [118], the matching task is formulated as an energy minimization problem and the objective function takes into account both feature descriptors and their spatial information. Dual decomposition is used to solve the minimization problem. The spectral matching [56] algorithm uses a spectral method where an adjacency matrix of a graph is defined with the nodes representing the potential correspondences and the weights on the links representing pairwise agreements between potential correspondences. The principal eigenvector of the matrix is used to recover the correct assignments based on how strongly they belong to the main cluster. Graph shift [62] method optimizes the same objective function as in [56] but with different constraints based on l_1 norm to find all large local maxima (including the global maximum) through a systematic way of initialization. A progressive graph matching framework is proposed in [16] where probabilistic progression and matching of graphs are combined to efficiently re-estimate the most plausible target graphs based on the current matching result. More recently, the vector field interpolation proposed in [66] uses an expectation-maximization algorithm to identify the implicit probabilistic model representing the uniformity that must be satisfied by optical flow derived from the true matches. Most methods in this type consider both feature similarity and spatial arrangement, and are relatively more robust to outliers than those using hypothesize-and-verify framework. However, solving a complex optimization problem can be NP-hard, making these algorithms inappropriate for robotics applications.

5.2.2 Keypoint Matching under Illumination Change

A main impact of illumination change on keypoint matching is the descriptor variance. Several attempts exist to neutralize the impact of illumination through improved descriptors. For example, a descriptor can be made partially invariant to illumination by choosing a proper color space in which to compute the descriptors. A comprehensive study in [50] evaluates the performance of color descriptors including SIFT in the application of scene recognition with illumination change. It is shown that color SIFT generally performs better than the original SIFT descriptor, which is computed on a grayscale image.

Alternatively, accurate keypoint matching can be achieved by learning a distance metric in the feature space where the correlations among the features are characterized in terms of the statistics of co-occurrence. This approach is based on the assumption that Euclidean distance may not be optimal in capturing the similarity between descriptors. An example of this approach is presented in [72] with its application to visual localization under illumination change in [94].

Yet another approach to address illumination invariance is through feature selection. The idea is to retain keypoints that are distinct, representative and easy to match, and eliminate the less reliable ones that impact matching negatively. Scale dependent feature selection [128] is one example with the observation that keypoints extracted at coarse scales usually provide better matching performance than those at fine scales, although the numbers are fewer. In general, keypoint matching repeatability can be learned and used to improve the result [43]. All techniques reviewed in this section are complementary to our outlier pruning algorithm, and they can be employed in combination with our method to obtain optimal keypoint matching performance.

5.3 Outlier Pruning with Consensus Constraint

This section describes our proposed outlier pruning algorithm. We start by introducing the weak perspective camera model, one of the assumptions that is used in our study. Subsequently, we discuss the statistics of optical flow that support our proposed algorithm. Then we explain how Chebyshev's inequality is used to prune outliers and, for that, we discuss how to determine the parameters needed for using the inequality.

5.3.1 Weak Perspective Camera Model and Optical Flow

The weak perspective camera model is a special case of a projective camera model when the depth of the scene points is small relative to their distance to the camera. A projective or pin-hole camera transforms a 3D scene point $\mathbf{X} = (X, Y, Z)^T$ into an image point $\mathbf{u} = (u, v)^T$ by

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{f}{Z} \begin{bmatrix} X \\ Y \end{bmatrix} \tag{5.1}$$

where f is the focal length of the camera. Perspective effect refers to the phenomenon that objects far away from the camera appear smaller than objects close by. This is clearly captured by the denominator Z in Equation (5.1). While this is the general case in most applications, there are instances where the perspective effect is small so that it can be ignored for computational advantages. Examples of *weak perspective projection* include when imaged objects are far from the camera relative to their depth, and when space points are on a planar surface almost parallel to the image plane. In such cases, the weak perspective camera model adequately models the projection process, and this gives rise to the first assumption in our study. In a weak perspective camera, all the geometrical properties in a pinhole camera, including the epipolar constraint hold true, and multi-view geometry is applicable. However, the transformation from a 3D scene to an image point is simplified to

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{f}{Z_{ave}} \begin{bmatrix} X \\ Y \end{bmatrix} \tag{5.2}$$

where Z_{ave} is the average distance of *all* the space points.

Note that in many robotics applications including loop closure detection, this weak perspective camera model can be assumed because most scene points are far from the camera especially in the outdoor environments, or for a subset of these points that have similar distances to the camera if they come from the same region of an object such as a building or a tree. While it is true that not all the scene points can be adequately modeled using weak perspective, the subset of points that do satisfy the weak perspective assumption can be handled by our method.

5.3.2 Flow Vector Distribution of Matching Keypoints

Under the weak perspective camera model, we will show in this section that the flow vectors of the matching keypoints follow an extremely simple distribution. It is this prior that we will use to prune putative matches for inliers. The flow vector of two matching keypoints in two images is determined by the positions of the 3D point with respect to the two camera positions when the 3D point is projected. Using the first camera as the reference frame, when a 3D space point is observed in the second camera, the position of the point with respect to the second camera is given by

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = R \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + L \tag{5.3}$$

where $(X, Y, Z)^T$ is the position of the 3D point in the first camera and R and $L = (L_X, L_Y, L_Z)^T$ are respectively the camera rotation and translation. For the application of loop closure detection of a ground vehicle as well as in many other applications, we can further assume locally planar motion, i.e., there can be only non-zero translations in *X* and *Z*, and a non-zero rotation θ around *Y* of the camera. In addition, with the weak perspective camera, Z can be replaced by Z_{ave}. The rotation and translation of the second camera become

$$R = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \tag{5.4}$$

$$L = (L_X, 0, L_Z)^T \tag{5.5}$$

As a result, the space point in the second camera is given by

$$X' = X\cos\theta + Z\sin\theta + L_X \tag{5.6}$$

$$Y' = Y \tag{5.7}$$

Then flow vector of the space point in the image plane is obtained by evaluating Equation (5.2) twice and then taking the difference

$$\begin{aligned} \Delta u &= \frac{f}{Z_{ave}}(X' - X) = \frac{f}{Z_{ave}} \cdot \frac{Z\sin\theta + L_X}{1 - \cos\theta} \\ &= \frac{f\sin\theta}{1 - \cos\theta} + \frac{fL_X}{Z_{ave}(1 - \cos\theta)} \end{aligned} \tag{5.8}$$

$$\Delta v = \frac{f}{Z_{ave}}(Y' - Y) = 0 \tag{5.9}$$

A quick inspection of Equation (5.8) and (5.9) reveals that both Δu and Δv are in fact independent of the 3D point. If we are to generate the distribution of the flow vectors of all the matching keypoints, then the distribution will be deterministic, at a single value with probability 1, and independent of the positions of the 3D points, i.e., under a weak perspective camera with planar motion, the induced optical flow of all the keypoints will be dependent only of the camera motion. Specifically, the horizontal (u) component is related to θ and L_X and v component is always 0. In practice, however, the weak perspective assumption is not strictly satisfied by all the space points, and there are errors also in keypoint locations, among others. As a result, the observed optical flow for matched keypoints follows a distribution whose mean is near that given by Equation (5.8) and (5.9), and whose dispersion is expected to be small. It is this peaked distribution that will serve as the basis for our outlier pruning algorithm and it is, not surprisingly, supported by previous studies on optical flow as will be briefly discussed in the next section.

5.3.3 Statistics of Optical Flow

The previous section derives the optical flow under planar camera motion for the points satisfying the weak perspective assumption. In fact, the statistics that serve as the basis of our outlier pruning method is related to the study in [97] where images in a video captured by a hand-held camera travelling forward are used to compute the horizontal and vertical components of optical flow, which are found experimentally to be approximately Laplacian, which is unimodal with a peaked form. However that result was obtained for multiple images, while our method focuses on a single image pair. In our case, although the distribution may not necessarily be Laplacian, it is expected to be unimodal when the weak perspective assumption holds true according to the analysis in the previous section. In a way, our study is consistent with the finding in [97] in terms of the simple statistics of the optical flow induced by planar camera motion. An obvious difference is that the distribution of optical flow from multiple image pairs was found to have a mode near 0 while in our case, it is in general not at 0 in the horizontal direction according to Equation (5.8). This is understandable because a statistical distribution built from multiple image pairs is a mixture of many distributions. Fortunately, our method depends only on the fact that matching keypoints generate a dominant peak in the distribution of optical flow vectors, and that the location of the mean is unimportant.

5.3.4 Chebyshev's Inequality

Although the flow vectors of the matching keypoints is shown to follow an extremely simple distribution under the assumptions of weak perspective camera and locally planar camera motion, the actual distribution of this displacement can be quite complex due to many reasons, such as non-planar camera motion, lens distortion, keypoint localization error, etc. More seriously, we are able to compute only the distribution of the flow vectors of *putative* matches, not that of the inlier matches. Therefore, in our algorithm, we must resort to a robust statistical tool to prune the putative matches. For this purpose, we use the Chebyshev's inequality as the basis to separate inliers from outliers among the putative matches. The inequality states that in any probability distribution, no more than $1/k^2$ of the distribution's values can be more than k standard deviations away from the mean (or equivalently, at least $1 - 1/k^2$ of the distribution's values are within k standard deviations of the mean). Mathematically, let $\mathbb{X}$ (integrable) be a random variable with finite expected value μ and finite non-zero variance σ^2. Then for any real number $k > 0$ (only the case $k > 1$ provides useful information),

$$\Pr(|\mathbb{X} - \mu| \geq k\sigma) \leq \frac{1}{k^2} \tag{5.10}$$

Equation (5.10) implies that for a given k, if σ is small, the interval to include a majority of data samples can be narrow. Therefore, the only requirement for our method to work is that the flow vector distribution is dominated by inliers near the mode and outliers do not contribute to a particular mode specifically. We can then estimate σ of the optical flow distribution using samples near the mode, and then select most inliers accordingly without including too many outliers. In fact, far more than $1-1/k^2$ inlier samples in our case are within the interval of $(\mu - k\sigma, \mu +$

$k\sigma)$, as will be shown in Section 5.4.

5.3.5 Consensus Constraint

Development in the previous sections has concluded that, under the weak perspective and planar motion assumptions, the distribution of keypoint displacement of inliers among the putative keypoint matches is unimodal and peaked at a mean independent of the 3D scene, in both u and v components. In this section, we will exploit this critical conclusion as a constraint on the inlier matches. We refer to this constraint as *consensus constraint*, to indicate the fact that the consistency in the flow vector displacement is a property that all inlier matches agree upon. As mentioned, for us to be able to take advantage of the consensus constraint, the displacement distribution of outliers must not distort this simple inlier distribution enough to skew the dominant peak in the combined distribution. While this is difficult to establish theoretically, one can consider the outlier distribution to result from random matches. Consequently, the distribution of the flow vector displacement due to outliers is similar to that of the distance between two random points in a rectangle defined by the image size. This distribution is also unimodal but with a large variance [91] and can be approximated by a uniform distribution [66] over an interval defined by the width and the height of the image. Therefore, it is reasonable to expect that the dominant peak of the inliers can be identified by a mode-seeking algorithm – for example *mean shift* in our case – to detect the mode in the mixture distribution. In summary, our algorithm for pruning the outliers using the consensus constraint can now be fully constructed as in Algorithm 5.1.

In Algorithm 5.1, mode($\mathcal{S}_p$) is to find the mode of the samples in $\mathcal{S}_p$ and var($\mathcal{S}_p$) is to calculate the variance, in both *x* and *y* components. In the last step, $\mathcal{S}_u \mapsto \mathcal{U}_{id}$ means mapping the elements in $\mathcal{S}_u$ to their matching *id*s, i.e., the number of i specified in Step 2. Therefore, the consensus set of inliers correspond to the samples that fall into the rectangular area bounded by $u_m \pm k\sigma_u$ and $v_m \pm k\sigma_v$. One issue is that in Chebyshev's inequality the mean of the distribution is used to determine the interval, while in our method we use the mode. We will show in the experiments that the estimated mode of all the samples is approximately the mean of inliers, with

Algorithm 5.1: Consensus Constraint for Keypoint Matching

```
Input  : Two sets of keypoint locations
         $\mathcal{S}_c = \{(\mathbf{u}_c, \mathbf{v}_c)\}$, $\mathcal{S}_q = \{(\mathbf{u}_q, \mathbf{v}_q)\}$
Output: Consensus set of inliers $\mathcal{I}$
1 Perform keypoint matching by nearest neighbor search and
  $(u_{c_i}, v_{c_i}) \leftrightarrow (u_{q_j}, v_{q_j})$ represents a matching pair, where $i \in \{1, 2, ..., N_c\}$,
  $j \in \{1, 2, ..., N_q\}$;
2 $\Delta u_i = u_{q_j} - u_{c_i}$, $\Delta v_i = v_{q_j} - v_{c_i}, \forall i \leq N_c$, $\mathcal{S}_p = \{(\Delta u_i, \Delta v_i) \ : \ \forall i \leq N_c\}$;
3 Run *mean shift* on $\mathcal{S}_p$, $(u_m, v_m) = \text{mode}(\mathcal{S}_p)$, $(\sigma_u^2, \sigma_v^2) = \text{var}(\mathcal{S}_p)$;
4 $\mathcal{S}_u = \{\Delta u_i \ : \ \forall \Delta u_i \in (u_m - k\sigma_u, u_m + k\sigma_u)\}$,
  $\mathcal{S}_v = \{\Delta v_i \ : \ \forall \Delta v_i \in (v_m - k\sigma_v, v_m + k\sigma_v)\}$;
5 $\mathcal{I} = (\mathcal{S}_u \mapsto \mathcal{U}_{id}) \cap (\mathcal{S}_v \mapsto \mathcal{V}_{id})$;
```

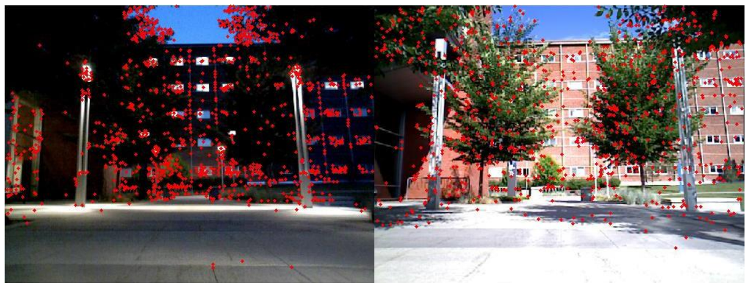
a distance of no more than a few pixels. Therefore, It can be used without affecting matching performance.

To ensure that mean shift can converge to the global maximum efficiently, we first build a 3D histogram by quantizing the 2D elements in $\mathcal{S}_p$. The mean value of the elements in the highest bin is used as the initialization in mean shift. This value can be expected to be close to the true mode, and the mode-seeking algorithm usually converges in a few steps with negligible computational cost. We use a flat kernel in mean shift so that all samples within the window size are considered equally important.

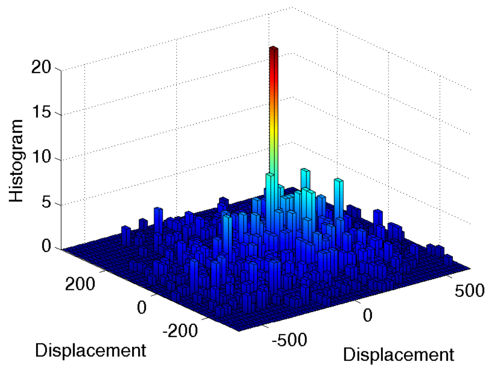
Figure 5.2 shows an example of the distributions of keypoint displacement. A matching image pair with all the keypoints are shown in 5.2a and the distribution of displacements after nearest neighbor matching is shown in 5.2b. 5.2c shows the surviving keypoints that correspond to the mode detected in 5.2d.

5.4 Experimental Results

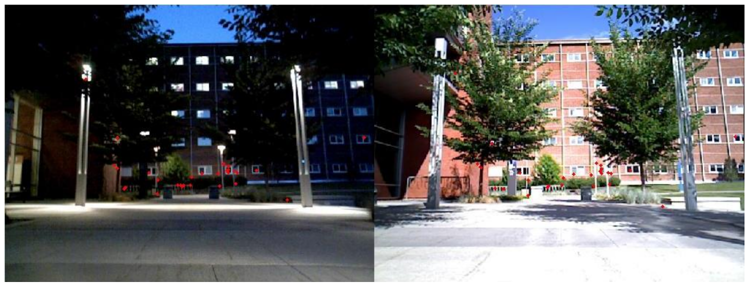
To evaluate our proposed keypoint matching method, we conducted comprehensive experiments on datasets involving various applications. For quantitative performance evaluation, we used the dataset popularized by Mikolajczyk and Schmid's seminal paper on performance evaluation of keypoint descriptors [71]. We refer to their dataset as the MS dataset. A main advantage of the MS dataset is the availability of the ground truth matches (inliers). Next, our algorithm was evaluated on



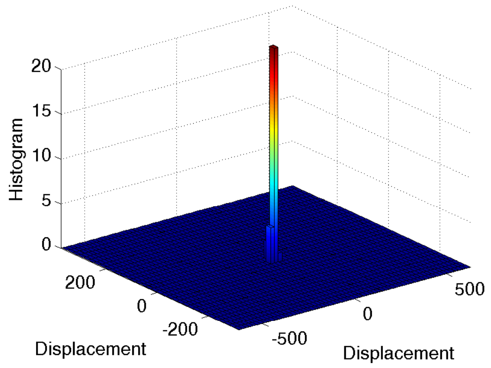
(a) All the keypoints in a matching image pair



(b) Distribution of displacements in 2D



(c) Surviving keypoints satisfying consensus constraint



(d) Distribution of displacements for surviving keypoints

Figure 5.2: An example of the distribution of keypoint displacements. The mode is detected by consensus constraint in (d) and the corresponding matching keypoint pairs are kept in (c).

a dataset we constructed in the application of visual loop closure detection under significant illumination change. Finally, we also tested our method qualitatively with images whose keypoints might need to be matched in the application of structure from motion. In our experiments, we compare four competing outlier pruning algorithms.

- **CC**. Our proposed algorithm to prune outliers by consensus constraint.
- **RANSAC**. Use of RANSAC and the epipolar constraint to prune outliers in nearest neighbor matches. In the application of loop closure verification, we used 2-point RANSAC [17] in which only two putative correspondences are required to estimate the camera motion parameters in the case of planar motion, to generate fair comparison to the proposed method.
- **Distance Ratio**. Default outlier pruning method proposed by Lowe to remove outliers by enforcing uniqueness.
- **VFC**. Vector field consensus algorithm that finds inliers through an optimization framework, considered as the state-of-the-art.

For keypoint detection and description, as well as nearest neighbor matching and RANSAC, we use OpenCV's implementation in the experiments. Implementation of our outlier pruning algorithm in both Matlab and Python is available online[1].

5.4.1 Quantitative Performance Evaluation

The MS dataset was used for quantitative evaluation. The dataset is a standard benchmark and has been popularly used in numerous studies involving performance evaluation of keypoint detection, description, and matching. The dataset contains eight image groups, each of six images of the same scene. Variance in each group is in either camera motion (rotation, translation, etc.) or image quality (blurring, compression, etc.). One of the six images was used as the reference image. Three groups, i.e., Bike (blur), Leuven (lighting change) and Trees (blur), are taken in conditions that are consistent with the assumptions in our method, and therefore

[1] `https://webdocs.cs.ualberta.ca/~liu17/`

Table 5.1: Number of Hypothetical Matches and Inlier Ratio

	#Hypothetical Matches	#Inliers	Inlier Ratio (%)
Bike 1-2		889	26.27
Bike 1-3		658	19.44
Bike 1-4	3384	387	11.44
Bike 1-5		293	8.66
Bike 1-6		196	5.79
Leuven 1-2		1271	51.65
Leuven 1-3		984	39.98
Leuven 1-4	2461	811	32.95
Leuven 1-5		704	28.61
Leuven 1-6		489	19.87
Trees 1-2		2730	20.57
Trees 1-3		2556	19.26
Trees 1-4	13269	1211	9.13
Trees 1-5		602	4.54
Trees 1-6		309	2.33

applicable in our study. Homography from the reference image to the other five in each group is given to facilitate building the ground truth matches reliably. In principle, two keypoints match if they satisfy the homography. In our experiment however, we further allow a small distance of a few pixels in both u and v in accepting a true match to account for the keypoint localization error. While changing this distance parameter may generate slightly different ground truth, as we have observed, it does not affect the comparison of different keypoint matching algorithms. Table 5.1 summarizes the number of putative matches and the inlier ratio for each image pair. Note that the number of putative matches after nearest neighbor matching is equivalent to the number of keypoints in the reference image. We can see that in terms of the inlier ratio, the group of Trees is the most challenging while Leuven is the simplest.

We first calculated the distance between the mode (u_m, v_m) estimated in Algorithm 5.1 and the mean of vector magnitudes from inliers for all the fifteen pairs. The results are summarized in Table 5.2. The mode and mean are close to each other with no more than a pixel in most cases. The maximum distance does not ex-

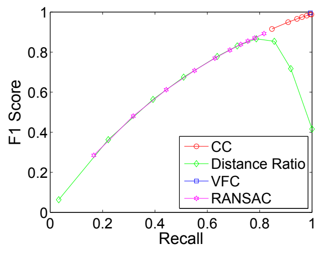
Table 5.2: Distance Between Mode and Mean in Pixels

Bike 1-2	0.1187	Leuven 1-2	0.0871	Trees 1-2	2.8022
Bike 1-3	0.2604	Leuven 1-3	0.0563	Trees 1-3	2.8135
Bike 1-4	0.4335	Leuven 1-4	0.1751	Trees 1-4	1.1337
Bike 1-5	0.1293	Leuven 1-5	0.1318	Trees 1-5	5.5997
Bike 1-6	0.4466	Leuven 1-6	0.2061	Trees 1-6	5.8252

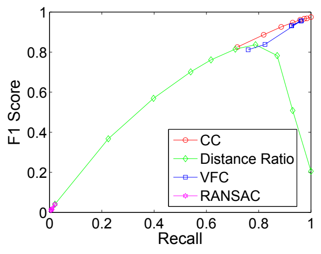
ceed six pixels. This confirms our assertion that it is reasonable to use the estimated mode from mean shift as the distribution mean required in Chebyshev's inequality in the outlier pruning process.

We then compared our method with the baseline RANSAC, distance ratio and vector field consensus (VFC) proposed in [66], the state-of-the-art algorithm with superior performance to previous algorithms such as graph shift [62] and MLE-SAC [116]. The implementation of VFC in Matlab by the authors of [66] was used. F_1 score and recall curve was used as the performance metric. We did not use precision as for some cases, VFC failed to compute a single match, and this would leave precision undefined. For RANSAC, distance ratio and the proposed method, F_1-recall curve was plotted as only one parameter varies in each case: the distance from a keypoint to the epipolar line in pixels, the distance ratio and *k*, respectively. Regarding VFC, multiple parameters may influence the performance. As suggested by the authors, we tuned two parameters in VFC: the inlier ratio and the bounds on the uniform distribution of the outliers, and then calculated the averaged F_1 score and recall.

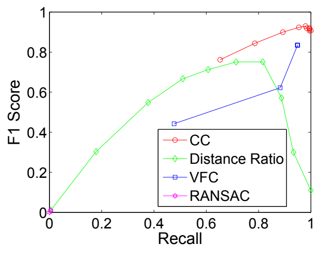
Nine of the fifteen groups of results are shown in Figure 5.3. In Bikes and Leuven, our method provides comparable and sometimes superior (see Bikes 1-6) performance to VFC. In addition, we emphasize that the performance in our method is more robust and insensitive to the parameter *k* as all the F_1-recall values are close. Therefore, we can also conclude that most inliers are included within a limited interval centered at the mode or mean. VFC occasionally generates poor results (see Trees 1-6) if the parameters do not favor the underlying model with respect to the optical flow distributions. The difficulty of finding such a model to reflect



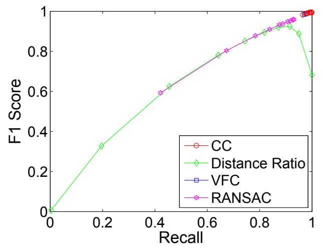
(a) Bikes 1-2



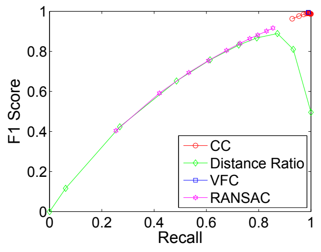
(b) Bikes 1-4



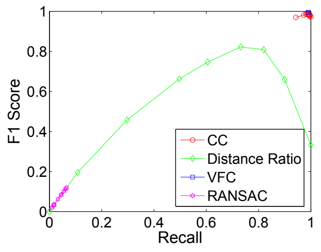
(c) Bikes 1-6



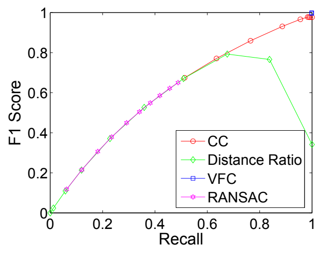
(d) Leuven 1-2



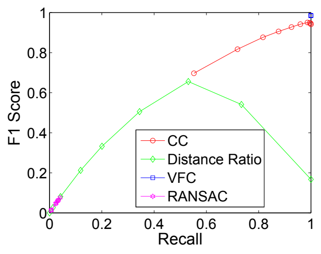
(e) Leuven 1-4



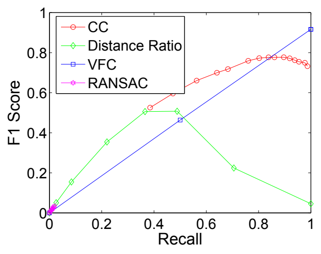
(f) Leuven 1-6



(g) Trees 1-2



(h) Trees 1-4



(i) Trees 1-6

Figure 5.3: F_1 score w.r.t. recall in the three image groups. In Bikes and Leuven, our method provides comparable or superior performance to VFC. VFC is highly sensitive to parameters and it may not find a single match with improper parameter settings (fails multiple times in Trees 1-6 as shown in the corner values). Distance ratio is generally acceptable and RANSAC can only work well when the inlier ratio is about 20% or higher.

the smoothness of flow from inliers in VFC can lie on the significant noise of the data, i.e., the low inlier ratio and indeterminate patterns of optical flow. A possible solution requires a strong prior of the distribution so that a good initialization can be achieved for correct convergence. In VFC however, such a prior is not fully exploited. On the other hand, our method makes use of the reasonable assumptions and investigates the distribution prior in identifying the model and the inliers.

VFC can outperform our proposed method under optimal parameter selection in the cases of Trees 1-2 and Trees 1-4. However, our method is already able to generate satisfactory results with above 50% recall at 80% precision. As in these examples there are hundreds of inliers, this recall number is sufficient in most vision and robotics applications. We should also mention that in Trees 1-6, VFC cannot find any match for some parameter settings, resulting in 0 recall, whereas our method can always find plenty of matches for all values of k.

RANSAC is not competitive with respect to the other algorithms in the comparative group. It performs well only with relatively larger inlier ratios of higher than 20%, although not always. Distance ratio performs better than RANSAC but worse than both VFC and the proposed method. However, its performance is in general stable. In other words, it is easy to choose a threshold *dr* that achieves a good trade-off between precision and recall. All images in the MS dataset involve relatively moderate illumination change, which is not challenging. In the next section, we will evaluate the performance of the algorithms using another more challenging dataset.

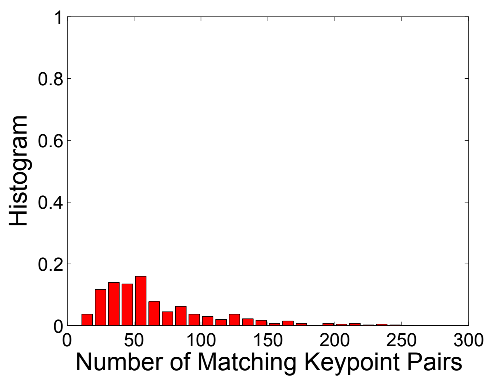
Finally, we emphasize that in all the methods, since nearest neighbor matching was first used to generate the putative matches, these algorithms are all considered pruning techniques to remove outliers. One can definitely apply RANSAC after any pruning method when necessary, but our goal is to show the effectiveness of the proposed method in outlier removal and therefore we omit the use of RANSAC after the pruning step.

5.4.2 Loop Closure Verification

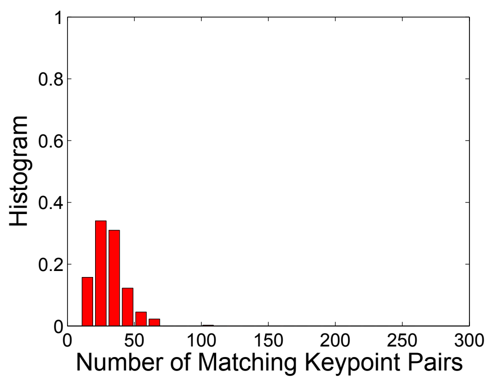
In the previous experiments, we dealt with only matching image pairs, i.e., the MS dataset has only image pairs in which true matches exist. The verification of a loop closure, however, requires a matching algorithm to distinguish successfully between the positive or matching case and the negative or non-matching case so that a verification decision can be made. To evaluate our method in verifying loop closures, we conducted experiments on the campus dataset that was used in the previous chapters.

As always, nearest neighbor matching is used to generate the initial putative keypoint matches for a given pair of images to be verified. Subsequently, a traditional method like RANSAC can be used to identify inliers. When illumination change is significant, for a positive image pair, the number of inliers among the putative matches goes down, and a low threshold on the number of putative matches that pass RANSAC has to be used, in order to verify the loop closure successfully. However, in the case of a pair of non-matching images where all putative matches are outliers, if a low threshold is used, RANSAC by chance could often find a camera motion between the two images with enough support. As a result, the verification decision process would consider the two images to be matching, incorrectly. The same issue happens with the VFC algorithm, i.e., VFC would often find an underlying motion model and accept negative image pairs. This implies that the trade-off between precision and recall of loop closure verification through MVG can be extremely difficult to make with lighting changes significantly. In contrast, our algorithm is able to use the consensus constraint to exclude most of the outliers first, for either a matching or a non-matching image pair without attempting to rely on RANSAC to find an underlying camera motion. In addition, to compare the proposed method fairly with RANSAC, we used 2-point algorithm [17] that also assumes planar camera motion as in our method. To better address the issue mentioned here, only the parameters reflecting small camera rotations (e.g. smaller than $\pi/12$) are considered and verified in the iteration of 2-point RANSAC.

In order to have an interesting comparison, we used the extreme case of matching between dark night and sunny day (see Figure 2.6d and 2.6e) as they are the most challenging situation of illumination change. We randomly selected 400 true matching image pairs and 400 non-matching pairs. Recall that a non-matching pair represents two images at two different locations that may still share similar local structures such as trees and buildings, and hence could be considered as a loop closure candidate by the detection algorithm. Again, we compare RANSAC, distance ratio test ($dr = 0.6$), VFC and our CC in generating inlier matches. To determine their performance, we calculated the histogram of the number of matches, for both the 400 positive and 400 negative cases, and the results are provided in Figures 5.4 through 5.7. Figure 5.4 and Figure 5.6 show that both RANSAC and VFC do not work well for negative cases since the number of matches for a non-matching image pair can be quite high. This makes it almost impossible to choose a threshold on the number of matches to separate the positive and negative cases without incur-



(a) Positive cases



(b) Negative cases

Figure 5.4: Histogram of the number of matches – RANSAC. The positive and negative cases are not distinguishable.

ring a major loss in recall or precision. The 2-point RANSAC does not essentially handle the problem of low inlier ratio. In fact, when the inlier ratio is low, an incorrect camera motion can get as many (or more) support as the true motion, and the thresholding of eliminating the large motion parameters does not significantly help, as two false matches could also generate small motion. Our proposed method, in contrast, significantly outperforms all the competing methods. For example, we can observe in Figure 5.7a that a majority of positive image pairs have enough matches, which is not the case in distance ratio method shown in Figure 5.5a. On the other hand, the proposed method can perform as well as distance ratio in identifying the negative cases by returning only a few matches as shown in Figure 5.7b.

To qualify the comparison, precision-recall curves are shown in Figure 5.8 where at 100% precision, consensus constraint surpasses distance ratio by 50% in the recall value. 2-point RANSAC and VFC are much worse than Distance Ratio and CC. For example, VFC is not able to reach 100% precision as there are about 25% negative cases that cannot be handled. Table 5.3 describes the average number of matching keypoints for the four methods. We can see that with significant illumination change, our proposed CC method finds four times as many matches as distance radio method, and also more matches than RANSAC after nearest neighbor matching.

To summarize, the main problem in distance ratio test is that it is purely based

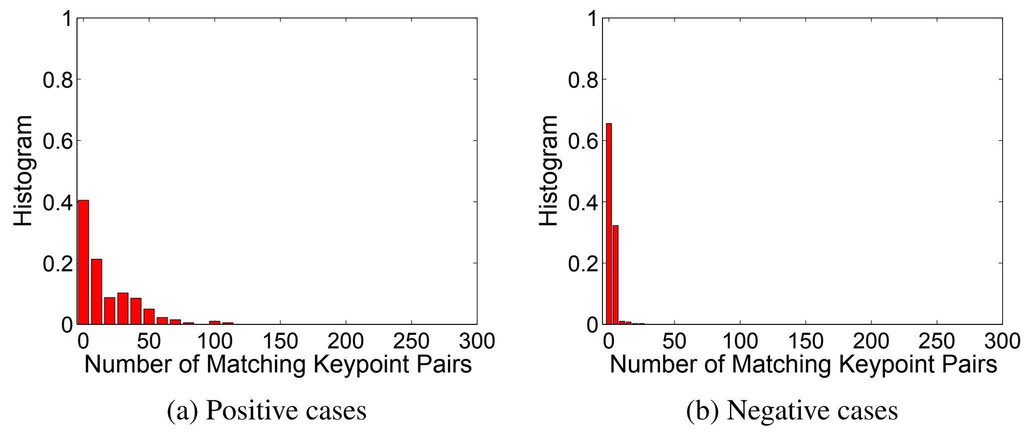


Figure 5.5: Histogram of the number of matches – distance ratio with $dr = 0.6$. 60% positive cases have no more than 10 matches due to significant illumination change.

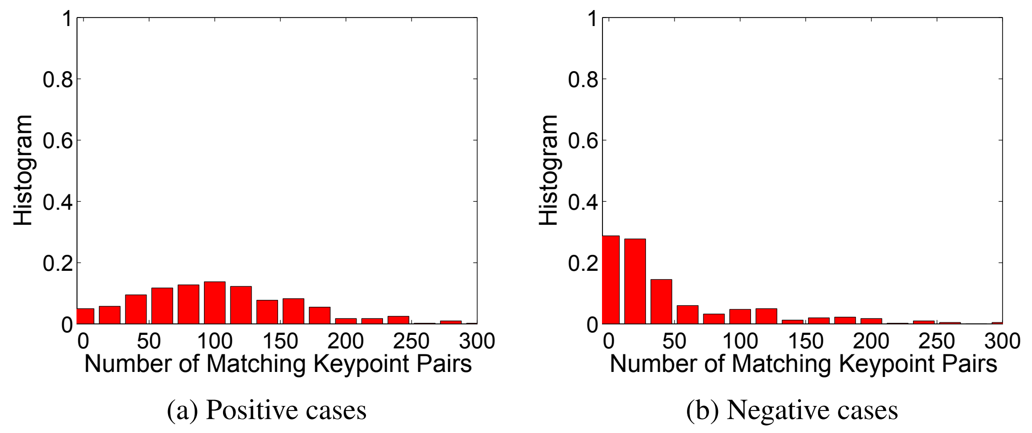
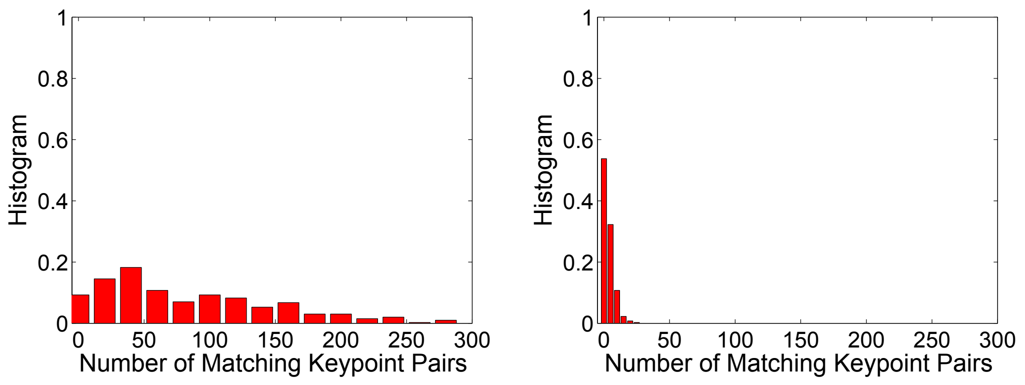


Figure 5.6: Histogram of the number of matches – VFC. Similar to RANSAC, the positive and negative cases are not distinguishable.

Table 5.3: Average Numbers of Matching Keypoints in Four Methods

	RANSAC	DR	VFC	CC
Positive	67	18	104	83
Negative	31	2	49	3



(a) Positive cases (b) Negative cases

Figure 5.7: Histogram of the number of matches – consensus constraint. Most positive cases have tens of or more matches, while a majority of negative cases have fewer than ten matches.

on similarity and therefore sensitive to descriptor variance which typically happens in illumination change. RANSAC is fragile to a large ratio of outliers as these incorrect matches can easily lead to a biased estimation of underlying camera motion. Similar situation applies to VFC, which optimizes a probabilistic model with unknown prior of the optical flow distributions and is therefore extremely sensitive to parameters. On the other hand, our method properly exploits the distribution prior and shows great improvement in the application of verifying loop closures.

5.4.3 Time Complexity

Time complexity is an important performance measure of an algorithm, especially in robotics application. Distance ratio test requires negligible extra time after nearest neighbor matching. The only potential time-consuming part in our method is the mode-seeking algorithm. However, since it is performed in 2D case with good initialization, the convergence to the mode usually takes only a few steps. Both RANSAC and VFC use iterative algorithms in finding the hypothetical model, and therefore their computational time is related to many factors. For example, the number of iterations for convergence in RANSAC depends on the number of data points from which the model can be instantiated, the percentage of outliers in the data points and the requested probability of success [33]. Since RANSAC has shown ob-

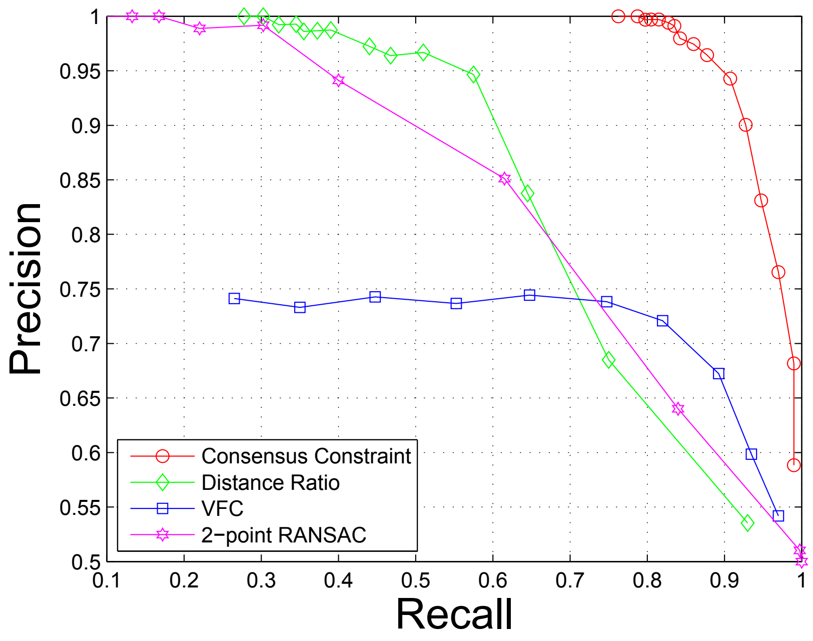


Figure 5.8: Precision-recall curves of the four methods in loop closure verification. Under significant illumination change, the proposed method significantly outperforms distance ratio test by 50% at 100% precision. Both methods are superior to RANSAC and VFC.

viously inferior performance to the proposed method, we only compare our method with VFC in computational time.

We investigated the relationship between the number of keypoints that need to be processed and the computational time, and used the speed-up factor, i.e., the ratio between the time of VFC and our algorithm, to describe the difference between the two methods. The experiments were conducted in the same hardware platform, and the VFC and CC pruning algorithms were both implemented in Matlab. The cases of Trees 1-2 and Trees 1-6 were used (20.57% and 2.33% inlier ratios respectively) and random subsets of different numbers of putative matches were uniformly sampled. For each number of keypoints, 100 runs were repeated and the average time was recorded for comparison. Figure 5.9 shows that our method is at least 20 time faster than VFC when more than 5000 keypoints need to be processed. The result is expected since VFC needs to solve an optimization problem whereas we only need to run mean shift. It is reasonable to believe that in general the time advantage of the proposed method will become more obvious when there is an increase in the number of keypoints.

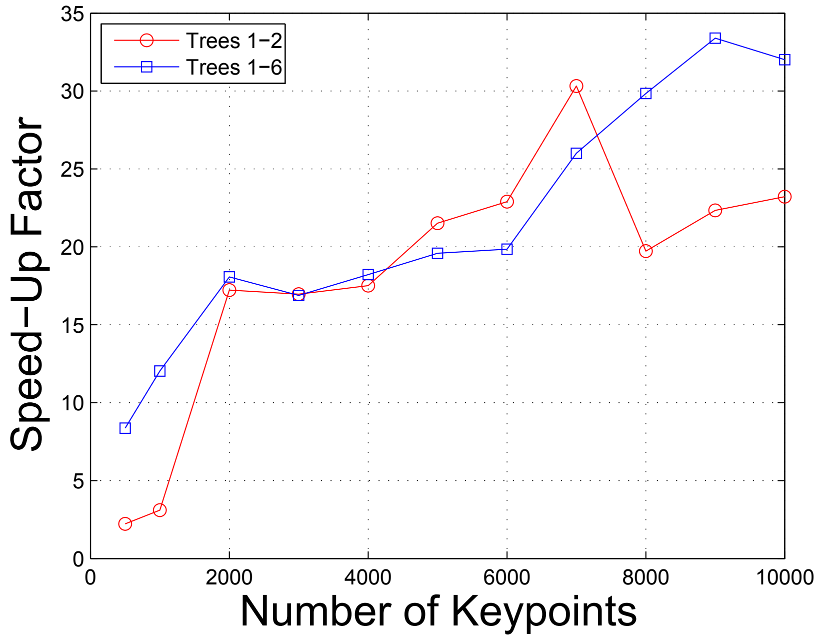


Figure 5.9: Speed-up factor vs. the number of keypoints. Our method is at least 20 time faster than VFC when there are more than 5000 keypoints that need to be processed.

5.5 Summary

Verification of loop closures is an important step in appearance SLAM. In this chapter, we have presented a simple but effective pruning method in order to match keypoints between two images to perform loop closure verification. Our algorithm is mainly based on the observations that in many applications one can make use of the weak perspective assumption and planar motion of the camera. In such a case, the inlier flow vectors of the matching keypoints are shown to be deterministic in both u and v components in their image coordinates. Therefore, this distribution prior can be exploited to remove outliers easily whose pixel displacements do not agree with the consensus, defined by the dominant mode of the vector flows of putative matches. Our method consists of a well-conditioned mode-seeking step plus an interval decision for inlier selection, and it is well supported by the related research on the statistics of optical flow. Experiments were conducted extensively on a standard benchmark dataset and outdoor image sequences with significant illumination change, for the purpose of evaluating our method in terms of both accuracy and efficiency in keypoint matching for loop closure verification. The results show that in the application of visual loop closure verification, the proposed algorithm outperforms the traditional outlier removal methods. It is also comparable or better than the the state-of-the-art VFC algorithm in most cases. Finally, our method is orders of magnitude more efficient than VFC, making it widely applicable in robotics applications. With the assistance of the proposed keypoint matching algorithm, loop closure verification can be easily achieved with high precision and recall – an important supplement to the SLAM algorithms in changing illumination environment proposed in the previous chapters.

$\sim$

Chapter 6

Conclusions

Appearance SLAM has become a main topic in robotics research and the challenging case of illumination change has attracted much attention in the recent studies. Focusing on several open problems in appearance SLAM such as perceptual aliasing, limitations in feature extraction and representation, and difficulty in geometric verification, this thesis studies appearance SLAM and place recognition in illumination changing environment. In this thesis, the case of illumination change becomes the mainline and around this line, efforts from several aspects have been made to address the related issues.

6.1 Achievements

In order to solve the problem of robot localization and mapping in changing illumination environment, the following achievements have been established.

- Direct feature matching is applied to address the problem of perceptual aliasing, a typical case in illumination change in appearance SLAM. To speed up the process of matching, we use a *k*-d tree to organize the raw features from the key frames in a topological map. Indexing visual features then becomes available with relevant key frames being retrieved as loop closure candidates. A verification can be used to further confirm the true matching location. We also apply scale dependent feature selection to reduce the number of features to exploit the scalability of the method. It is shown that feature matching with a *k*-d tree can significantly improve the recall of loop closure detection in a moderate size environment and therefore is applicable in alleviating the problem of illumination change in such a case.

- In order to deal with the case of significant illumination change in a large environment, we propose to use a whole-image descriptor – Gabor Gist, plus a novel application of the probabilistic framework. The proposed Bayes filtering exploits the MCMC approach, particularly, a particle filter to sample the state space of robot locations and capture the temporal coherence in an image sequence to ensure the detection continuity. PCA is used to reduce the dimensionality of the original descriptor and improve the discriminating power. The method is highly scalable due to the compactness of the descriptor and simplicity of the particle filter, and applicable in large maps. In addition, it performs even better than the feature-based approaches and can be easily extended in conjunction with a sequence-based matching technique to further improve its capability in handling illumination change.

- Performance evaluation of several types of whole-image descriptors are provided. Our study is conducted in the context of multiple robot applications, including loop closure detection and global localization of the kidnapped robot problem. By analyzing the performance in terms of both accuracy and processing time, we provide an important guideline of selecting these descriptors in related applications that can benefit from a compact and discriminative image descriptor.

- A simple and reliable keypoint matching algorithm is proposed for the last step of verification in appearance SLAM. Different from most of the existing methods that depend solely on the feature descriptors and therefore fail in the case of illumination change where both descriptor variance and keypoint repeatability become issues, our method exploits the spatial constraint of flow vectors from matching keypoints and uses it as a prior to perform outlier pruning. The method is developed based on the weak perspective camera model and planar motion that hold true in most robot and vision applications. Therefore, it is not limited to loop closure verification, but can be also generalized to other problems such as structure from motion (SfM) and visual odometry.

6.2 Future Research

Based on the existing achievements and results in this thesis, future research in appearance SLAM and place recognition can be identified as follows.

- Incorporating feature matching in filtering framework. Our proposed method in Chapter 2 does not exploit the temporal coherence in an image sequence and therefore considers loop closure detection simply an image retrieval problem. Clearly, important information of detection continuity can be exploited here. We would like to add the probabilistic localization framework in the method in our future work. As indexing is used in feature matching, the MCMC approach is not applicable. However, the work in [1, 2] is still an important reference and can be possibly used in our method. In addition, increasing the accuracy of feature matching is another research issue.

- Developing and extending other whole-image descriptors. We have conducted performance evaluation on several whole-image descriptors classified into multiple types. However, more such kind of descriptors need to be validated. It has been shown in [126] that building a map with 20 million key locations is possibly with a binary descriptor derived from a downsampled image. Similarly in [74], the author claimed that a map of the world can be possibly loaded to a normal storage device, thanks to the highly compact binary descriptors. It is extremely meaningful if such an image descriptor can work in the case of significant illumination change. In addition, there have been several instances of using the popular convolutional neural network (CNN) in developing descriptors that are applicable in place recognition [15, 110]. The most obvious advantage is that no training is needed as public CNNs are available for scene and object recognition.

- Localization and mapping across season. In addition to illumination change addressed in this thesis, seasonal change is also an important type of dynamic change in long-term autonomy. It is shown in [122] that local invariant features such as SIFT and SURF are able to deal with seasonal change at a certain level. The assumptions are however, high resolution images are required and feature matching should be used as a similarity metric. Alternatively, a whole-image descriptor can be used in learning the change as described in [108]. In fact, the challenge in the case of SLAM across seasons lies in the scene change, for example, between summer and winter. Therefore, how to identify and encode the invariant component in the scene is the key to solving the problem.

∼

Bibliography

[1] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat, “Real-time visual loop-closure detection,” in *IEEE International Conference on Robotics and Automation*, Pasadena, USA, 2008, pp. 1842–1847.

[2] 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, pp. 1027–1037, 2008.

[3] H. Badino, D. Huber, and T. Kanade, “Real-time topometric localization,” in *IEEE International Conference on Robotics and Automation*, St. Paul, MN, USA, 2012, pp. 1635–1642.

[4] T. Bailey and H. Durrant-Whyte, “Simultaneous localisation and mapping (SLAM): Part II,” *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, pp. 108–117, 2006.

[5] A. Baumberg, “Reliable feature matching across widely separated views,” in *IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head, SC, USA, 2000, pp. 1774–1781.

[6] H. Baya, A. Essa, T. Tuytelaars, and L. V. Gool, “Speeded-up robust features (SURF),” *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 346–359, 2008.

[7] J. Beis and D. Lowe, “Shape indexing using approximate nearest-neighbour search in high-dimensional spaces,” in *IEEE Conference on Computer Vision and Pattern Recognition*, San Juan, Puerto Rico, 1997, pp. 1000–1006.

[8] F. Bonin-Font, P. L. N. Carrasco, A. B. Burguera, and G. O. Codina, "LSH for loop closing detection in underwater visual SLAM," in *IEEE International Conference on Emerging Technology and Factory Automation*, Barcelona, Spain, 2014.

[9] S. Bonnabel, P. Martin, and E. Salaün, "Invariant extended kalman filter: theory and application to a velocity-aided attitude estimation problem," in *IEEE Conference on Decision and Control*, Shanghai, China, 2009, pp. 1297–1304.

[10] S. Borthwick and H. Durrant-Whyte, "Simultaneous localisation and map building for autonomous guided vehicles," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Munich, Germany, 1994, pp. 761–768.

[11] J. Callmer, K. Granström, J. Nieto, and F. Ramos, "Tree of words for visual loop closure detection in urban SLAM," in *Australasian Conference on Robotics and Automation*, Canberra, Australia, 2008.

[12] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary robust independent elementary features," in *European Conference on Computer Vision*, Heraklion, Crete, Greece, 2010, pp. 778–792.

[13] J. Canny, "A computational approach to edge detection," *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–714, 1986.

[14] N. Carlevaris-Bianco and R. M. Eustice, "Learning visual feature descriptors for dynamic lighting conditions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Chicago, IL, 2014, pp. 2769–2776.

[15] Z. Chen, O. Lam, A. Jacobson, and M. Milford, "Convolutional neural network-based place recognition," in *Australasian Conference on Robotics and Automation*, Melbourne, Australia, 2014.

[16] M. Cho and K. M. Lee, "Progressive graph matching: Making a move of graphs via probabilistic voting," in *IEEE Conference on Computer Vision and Pattern Recognition*, Providence, Rhode Island, 2012, pp. 398–405.

[17] C. Chou and C. Wang, "2-point ransac for scene image matching under large viewpoint changes," in *IEEE International Conference on Robotics and Automation*, Seattle, USA, 2015, pp. 3646–3651.

[18] O. Chum and J. Matas, "Matching with prosac - progressive sample consensus," in *IEEE Conference on Computer Vision and Pattern Recognition*, Beijing, China, 2005, pp. 220–226.

[19] O. Chum, J. Matas, and J. Kittle, "Locally optimized ransac," *Pattern Recognition*, vol. 2781, pp. 236–243, 2003.

[20] O. Chum, M. Perdoch, and J. Matas, "Geometric min-hashing: Finding a (thick) needle in a haystack," in *IEEE Conference on Computer Vision and Pattern Recognition*, Miami, USA, 2009, pp. 17–24.

[21] O. Chum, J. Philbin, and A. Zisserman, "Near duplicate image detection: min-hash and tf-idf weighting," in *British Machine Vision Conference*, Leeds, UK, 2008.

[22] W. Churchill and P. Newman, "Experience-based navigation for long-term localisation," *International Journal of Robotics Research*, vol. 32, pp. 1645–1661, 2013.

[23] M. Cummins and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *International Journal of Robotics Research*, vol. 27, pp. 647–665, 2008.

[24] ——, "Highly scalable appearance-only SLAM–FAB-MAP 2.0," in *Robotics: Science and Systems*, Seattle, USA, 2009.

[25] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, USA, 2005, pp. 886–893.

[26] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, “Locality-sensitive hashing scheme based on p-stable distributions,” in *Twentieth Annual Symposium on Computational Geometry*, Brooklyn, USA, 2004, pp. 253–262.

[27] A. Davision, I. Reid, N. Molton, and O. Stasse, “MonoSLAM: Real-time single camera slam,” *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[28] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, “Monte carlo localization for mobile robots,” in *IEEE International Conference on Robotics and Automation*, Detroit, USA, 1999, pp. 1322–1328.

[29] H. Deng, W. Zhang, E. Mortensen, T. Dietterich, and L. Shapiro, “Principal curvature-based region detector for object recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition*, Minneapolis, USA, 2007.

[30] A. Doucet, N. D. Freitas, and N. Gordon, “An introduction to sequential monte carlo methods,” in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. D. Freitas, and N. Gordon, Eds. Springer-Verlag, 2001, pp. 3–14.

[31] A. Doucet, N. D. Freitas, K. Murphy, and S. Russell, “Rao-Blackwellised particle filtering for dynamic bayesian networks,” in *Conference on Uncertainty in Artificial Intelligence*, Stanford, USA, 2000, pp. 176–183.

[32] H. Durrant-Whyte and T. Bailey, “Simultaneous localisation and mapping (SLAM): Part I the essential algorithms,” *IEEE Robotics & Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[33] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communication of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[34] D. Fox, S. Thrun, W. Burgard, and F. Dellaert, “Particle filters for mobile robot localization,” 2001.

[35] D. Gálvez-López and J. D. Tardós, “Real-time loop detection with bags of binary words,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, USA, 2011, pp. 1234–1241.

[36] ——, “Bags of binary words for fast place recognition in image sequences,” *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[37] R. G. Golledge, “Place recognition and wayfinding: Making sense of space,” *Geoforum*, vol. 23, pp. 199–214, 1992.

[38] J. Guivant and E. Nebot, “Optimization of the simultaneous localization and map building algorithm for real-time implementation,” *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 242–257, 2001.

[39] ——, “Improving computational and memory requirements of simultaneous localization and map building algorithms,” in *International Conference on Robotics and Automation*, Washington, DC, USA, 2002, pp. 2371–2376.

[40] K. Hajebi, Y. Abbasi-Yadkori, H. Shahbazi, and H. Zhang, “Fast approximate nearest-neighbor search with k-nearest neighbor graph,” in *International Joint Conferences on Artificial Intelligence*, Barcelona, Spain, 2011.

[41] K. Hajebi and H. Zhang, “An efcient index for visual search in appearance-based SLAM,” in *IEEE International Conference on Robotics and Automation*, Hong Kong, China, 2014, pp. 353–358.

[42] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[43] W. Hartmann, M. Havlena, and K. Schindler, “Predicting matchability,” in *EEE Conference on Computer Vision and Pattern Recognition*, Columbus, Ohio, 2014, pp. 9–16.

[44] J. Heinly, E. Dunn, , and J. Frahm, “Comparative evaluation of binary features,” in *European Conference on Computer Vision*, Firenze, Italy, 2012, pp. 759–773.

[45] G. Huang, A. Mourikis, and S. Roumeliotis, “Observability-based rules for designing consistent ekf SLAM estimators,” *International Journal of Robotics Research*, vol. 29, no. 5, pp. 502–528, 2010.

[46] E. Johns and G.-Z. Yang, “Dynamic scene models for incremental, long-term, appearance-based localisation,” in *IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, 2013, pp. 2731–2736.

[47] ——, “Feature co-occurrence maps: Appearance-based localisation throughout the day,” in *IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, 2013, pp. 3212–3218.

[48] S. Julier and J. Uhlmann, “A new extension of the kalman filter to nonlinear systems,” in *International Synonym on Aerospace/Defense Sensing, Simulation, and Controls*, Orlando, Florida, 1997, pp. 182–193.

[49] S. Julier, J. Uhlmann, and H. Durrant-Whyte, “A new method for the nonlinear transformation of means and covariances in filters and estimators,” *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 477–482, 2000.

[50] T. G. K. E. A. van de Sande and C. G. M. Snoek, “Evaluating color descriptors for object and scene recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 1582–1596, 2010.

[51] T. Kadir and M. Brady, “Saliency, scale and image description,” *International Journal of Computer Vision*, vol. 45, no. 2, pp. 83–105, 2001.

[52] T. Kadir, A. Zisserman, and M. Brady, “An affine invariant salient region detector,” in *European Conference on Computer Vision*, Prague, Czech Republic, 2004, pp. 228–241.

[53] A. Kawewong, N. Tongprasit, S. Tangruamsub, and O. Hasegawa, “Online and incremental appearance-based SLAM in highly dynamic environments,” *International Journal of Robotics Research*, vol. 30, no. 1, pp. 33–55, 2011.

[54] Y. Ke and R. Sukthankar, “PCA-SIFT: a more distinctive representation for local image descriptors,” in *IEEE Conference on Computer Vision and Pattern Recognition*, Washington, DC, USA, 2004, pp. 506–513.

[55] S. Lazebnik, C. Schmid, and J. Ponce, “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories,” in *IEEE Conference on Computer Vision and Pattern Recognition*, New York, USA, 2006, pp. 2169–2178.

[56] M. Leordeanu and M. Hebert, “A spectral technique for correspondence problems using pairwise constraints,” in *IEEE International Conference on Computer Vision*, Beijing, China, 2005, pp. 1482–1489.

[57] S. Leutenegger, M. Chli, and R. Siegwart, “BRISK: Binary robust invariant scalable keypoints,” in *IEEE International Conference on Computer Vision*, Barcelona, Spain, 2011, pp. 2548–2555.

[58] T. Lindeberg, “Feature detection with automatic scale selection,” *International Journal of Computer Vision*, vol. 30, no. 2, pp. 79–116, 1998.

[59] ——, “Image matching using generalized scale-space interest points,” in *Scale Space and Variational Methods in Computer Vision*, A. Kuijper, K. Bredies, T. Pock, and H. Bischof, Eds. Springer Berlin Heidelberg, 2013, pp. 355–367.

[60] ——, “Scale selection properties of generalized scale-space interest point detectors,” *Journal of Mathematical Imaging and Vision*, vol. 46, no. 2, pp. 177–210, 2013.

[61] T. Lindeberg and J. Garding, “Shape-adapted smoothing in estimation of 3-d shape cues from affine distortions of local 2-d brightness structure,” *Image and Vision Computing*, vol. 15, no. 6, pp. 415–434, 1997.

[62] H. Liu and S. Yan, “Common visual pattern discovery via spatially coherent correspondences,” in *IEEE International Conference on Computer Vision*, Beijing, China, 2005, pp. 1609–1616.

[63] J. S. Liu and R. Chen, “Sequential monte carlo methods for dynamic systems,” *Journal of the American Statistical Association*, vol. 93, pp. 1032–1044, 1998.

[64] D. Lowe, “Object recognition from local scale-invariant features,” in *IEEE International Conference on Computer Vision*, Corfu, Greece, 1999, pp. 1150–1157.

[65] ——, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[66] J. Ma, J. Zhao, J. Tian, A. Yuille, and T. Tu, “Robust point matching via vector field consensus,” *IEEE Transactions on Image Processing*, vol. 23, no. 4, pp. 1706–1721, 2014.

[67] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide baseline stereo from maximally stable extremal regions,” in *British Machine Vision Conference*, Cardiff, UK, 2002, pp. 384–396.

[68] K. Mikolajczyk and C. Schmid, “An affine invariant interest point detector,” in *European Conference on Computer Vision*, Copenhagen, Denmark, 2002, pp. 128–142.

[69] ——, “Scale & affine invariant interest point detectors,” *International Journal of Computer Vision*, vol. 60, no. 1, pp. 63–86, 2004.

[70] ——, “A performance evaluation of local descriptors,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.

[71] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, and *et. al.*, "A comparison of affine region detectors," *International Journal of Computer Vision*, vol. 65, no. 1–2, pp. 43–72, 2005.

[72] A. Mikulík, M. Perdoch, O. Chum, and J. Matas, "Learning a fine vocabulary," in *European Conference on Computer Vision*, Crete, Greece, 2010.

[73] M. Milford, "Visual route recognition with a handful of bits," in *Robotics: Science and Systems*, Sydney, Australia, 2012.

[74] ——, "Vision-based place recognition: how low can you go?" *International Journal of Robotics Research*, vol. 32, no. 7, pp. 766–789, 2013.

[75] M. Milford and A. Jacobson, "Brain-inspired sensor fusion for navigating robots," in *IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, 2013, pp. 2906–2913.

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

[77] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using FastSLAM," in *IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, 2003, pp. 1985–1991.

[78] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002, pp. 593–589.

[79] ——, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *International Joint Conference on Artificial Intelligence*, Acapulco, Mexico, 2003, pp. 1151–1156.

[80] M. Muja and D. Lowe, “Fast approximate nearest neighbors with automatic algorithm configuration,” in *International Conference on Computer Vision Theory and Applications*, Lisboa, Portugal, 2009.

[81] ——, “Fast matching of binary features,” in *Canadian Conference on Computer and Robot Vision*, Toronto, Canada, 2012, pp. 404–410.

[82] ——, “Scalable nearest neighbor algorithms for high dimensional data,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2227–2240, 2014.

[83] R. Mur-Artal and J. D. Tardós, “Fast relocalisation and loop closing in keyframe-based SLAM,” in *International Conference on Robotics and Automation*, Hong Kong, USA, 2014, pp. 846–853.

[84] A. C. Murillo, G. Singh, J. Košecká, and J. Guerrero, “Localization in urban environments using a panoramic gist descriptor,” *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 146–160, 2013.

[85] K. Murphy, “Bayesian map learning in dynamic environments,” in *Advances in Neural Information Processing Systems*, Denver, USA, 2000, pp. 1015–1021.

[86] P. Newman, D. Cole, and K. Ho, “Outdoor SLAM using visual appearance and laser ranging,” in *IEEE International Conference on Robotics and Automation*, Orlando, USA, 2006, pp. 1180–1187.

[87] D. Nistér and H. Stewénius, “Scalable recognition with a vocabulary tree,” in *IEEE Conference on Computer Vision and Pattern Recognition*, New York, USA, 2006, pp. 2161–2168.

[88] 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.

[89] E. Pepperell, P. Corke, and M. Milford, "All-environment visual place recognition with smart," in *IEEE International Conference on Robotics and Automation*, Hong Kong, China, 2014, pp. 1612–1618.

[90] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, Minneapolis, USA, 2007.

[91] J. Philip, "The probability distribution of the distance between two random points in a box," *TRITA MAT*, vol. 7, no. 10, 2007.

[92] A. Pronobis, B. Caputo, P. Jensfelt, and H. Christensen, "A discriminative approach to robust visual place recognition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 2006, pp. 3829–3836.

[93] R. Raguram, J. M. Frahm, and M. Pollefeys, "A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus," in *European Conference on Computer Vision*, Marseille, France, 2008, pp. 500–513.

[94] A. Ranganathan, S. Matsumoto, and D. Ilstrup, "Towards illumination invariance for visual localization," in *IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, 2013, pp. 3791–3798.

[95] D. Rodriguez-Losada and F. Matia, "Integrating segments and edges in featured-based SLAM," in *International Conference on Advanced Robotics*, Coimbra, Portugal, 2003, pp. 1717–1722.

[96] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European Conference on Computer Vision*, Graz, Austria, 2006, pp. 430–443.

[97] S. Roth and M. J. Black, "On the spatial statistics of optical flow," in *IEEE International Conference on Computer Vision*, Beijing, China, 2005, pp. 42–49.

[98] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to sift or surf," in *IEEE International Conference on Computer Vision*, Barcelona, Spain, 2011, pp. 2564–2571.

[99] H. Shahbazi and H. Zhang, "Application of locality sensitive hashing to realtime loop closure detection," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, USA, 2011, pp. 1228–1233.

[100] C. Siagian and L. Itti, "Rapid biologically-inspired scene classification using features shared with visual attention," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 300–312, 2007.

[101] ——, "Biologically inspired mobile robot vision localization," *IEEE Transactions on Robotics*, vol. 25, no. 4, pp. 861–873, 2009.

[102] C. Silpa-Anan and R. Hartley, "Optimised KD-trees for fast image descriptor matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, Anchorage, USA, 2008.

[103] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *IEEE International Conference on Computer Vision*, Nice, France, 2003, pp. 1470–1477.

[104] R. Smith and P. Cheesman, "On the representation and estimation of spatial uncertainty," *Internation Journal of Robotics Research*, vol. 5, no. 4, 1986.

[105] R. Smith, M. Self, and P. Cheesman, "Estimating uncertain spatial relationships in robotics," in *Autonomous Robot Vehicles*, I. J. Cox and G. T. Wilfong, Eds. Springer-Verlag, 1990, pp. 167–193.

[106] S. Smith and J. Brady, "Susan–a new approach to low level image processing," *International Journal of Computer Vision*, vol. 23, no. 1, pp. 45–78, 1997.

[107] N. Sünderhauf, P. Neubert, and P. Protzel, "Are we there yet? challenging seqSLAM on a 3000 km journey across all four seasons," in *Workshop on IEEE*

International Conference on Robotics and Automation, Karlsruhe, Germany, 2013.

[108] ——, “Predicting the change – a step towards life-long operation in everyday environments,” in *Workshop on Robotics: Science and Systems*, Berlin, Germany, 2013.

[109] N. Sünderhauf and P. Protzel, “BRIEF-Gist – Closing the loop by simple means,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, USA, 2011, pp. 1234–1241.

[110] N. Sünderhauf, S. Shirazi, A. Jacobson, F. Dayoub, E. Pepperell, and *et. al.*, “Place recognition with convnet landmarks: Viewpoint-robust, condition-robust, training-free,” in *Robotics: Science and Systems*, Rome, Italy, 2015.

[111] S. Thrun, W. Burgard, and D. Fox, “A probabilistic approach to concurrent mapping and localization for mobile robots,” *Machine Learning*, vol. 31, no. 1–3, pp. 29–53, 1998.

[112] ——, “A probabilistic approach to concurrent mapping and localization for mobile robots,” *Autonomous Robots*, vol. 5, no. 3–4, pp. 253–271, 1998.

[113] ——, “A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping,” in *IEEE International Conference on Robotics and Automation*, San Francisco, USA, 2000, pp. 321–328.

[114] ——, *Probabilistic Robotics.* Cambridge, MA: The MIT Press, 2005, p. 252.

[115] S. Thrun, Y. Liu, A. Ng, Z. Ghahramani, and H. Durrant-Whyte, “Simultaneous localization and mapping with sparse extended information filters,” *International Journal of Robotics Research*, vol. 23, no. 7, pp. 693–716, 2004.

[116] P. H. S. Torr and A. Zisserman, “MLESAC: A new robust estimator with application to estimating image geometry,” *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[117] A. Torralba, K. P. Murphy, W. T. Freeman, and M. A. Rubin, "Context-based vision system for place and object recognition," in *IEEE International Conference on Computer Vision*, Nice, France, 2003.

[118] L. Torresani, V. Kolmogorov, and C. Rother, "Feature correspondence via graph matching: Models and global optimization," in *European Conference on Computer Vision*, Marseille, France, 2008, pp. 596–609.

[119] T. Tuytelaars and L. V. Gool, "Wide baseline stereo matching based on local, afnely invariant regions," in *British Machine Vision Conference*, Bristol, UK, 2000, pp. 412–425.

[120] ——, "Matching widely separated views based on affine invariant regions," *International Journal of Computer Vision*, vol. 59, no. 1, pp. 63–86, 2004.

[121] I. Ulrich and I. Nourbakhsh, "Appearance-based place recognition for topological localization," in *IEEE International Conference on Robotics and Automation*, San Francisco, USA, 2000, pp. 1023–1029.

[122] C. Valgren and A. J. Lilienthal, "Sift, surf & seasons: Appearance-based long-term localization in outdoor environments," *Robotics and Autonomous Systems*, vol. 58, pp. 149–156, 2010.

[123] M. Walter, R. Eustice, and J. Leonard, "A provably consistent method for imposing sparsity in feature-based slam information filters," in *International Symposium on Robotics Research*, San Francisco, USA, 2005, pp. 214–234.

[124] X. Wang and H. Zhang, "Good image features for bearing-only SLAM," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 2008, pp. 2576–2581.

[125] J. Wu and J. M. Rehg, "CENTRIST: A visual descriptor for scene categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1489–1501, 2011.

[126] J. Wu and H. Zhang, "An efficient visual loop closure detection method in a map of 20 million key locations," in *IEEE International Conference on Robotics and Automation*, Hong Kong, China, 2014, pp. 861–866.

[127] B. Yamauchi and P. Langley, "Place recognition in dynamic environments," *Journal of Robotics Systems*, vol. 14, pp. 107–120, 1997.

[128] H. Zhang, "BoRF: Loop-closure detection with scale invariant visual features," in *IEEE International Conference on Robotics and Automation*, Shanghai, China, 2011, pp. 3125–3130.

[129] H. Zhang, B. Li, and D. Yang, "Keyframe detection for appearance-based visual SLAM," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, Taiwan, 2010, pp. 2071–2076.

[130] J. Zobel, A. Moffat, and K. Ramamohanarao, "Inverted files versus signature files for text indexing," *ACM Transactions on Database Systems*, vol. 23, no. 4, pp. 453–490, 1998.

∼