View-based mapping for wheeled robots

View-based mapping for wheeled robots

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. D.C. van den Boom
ten overstaan van een door het college voor promoties ingestelde
commissie, in het openbaar te verdedigen in de Agnietenkapel
op vrijdag 25 november 2011, te 10:00 uur

door

Olaf Booij

geboren te Leiderdorp

Promotiecommissie

Promotor:	Prof. dr. ir. F.C.A. Groen
Co-promotor:	Prof. dr. ir. B.J.A. Kröse
Overige leden:	Prof. dr. D.M. Gavrila
	Prof. dr. ir. R.L. Lagendijk
	Dr. K. Schutte
	Prof. dr. R. Siegwart
	Dr. ir. L. Dorst

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

This work has been partially supported by the EU Integrated Project COGNIRON (The Cognitive Robot Companion).

This work was carried out in the ASCI graduate school.
ASCI dissertation series number 246.

Copyright © 2011 by Olaf Booij
This work is licensed under the Creative Commons Attribution 3.0 NL licence `http://creativecommons.org/licenses/by/3.0/nl/deed.en_GB`.

Contents

1 Introduction

Traditionally, robots are deployed in industrial environments for automatic assembly or transportation tasks. These environments have well defined structured spatial layouts and can be augmented with artificial beacons. Spatial knowledge, which the robot might need, can be given prior to performing its tasks or can be easily derived from the beacons.

In recent years, robots have been deployed in public places and in homes among people. Examples are robot companions, office delivery robots and the successful autonomous vacuum cleaner (see Figure 1.1). These mobile robots have to operate in human inhabited environments, of which the structure is unknown prior to deployment. Current commercial robot products, such as autonomous vacuum cleaners, have a very limited notion of space, and drive around more or less randomly using simple bumper sensors. This suffices for cleaning a single room, but for multiple rooms extra equipment - such as beacons positioned in the door opening - is needed. Moreover, it is impossible for the robot to prevent getting entangled by power cords lying on the floor or to avoid bumping into people.

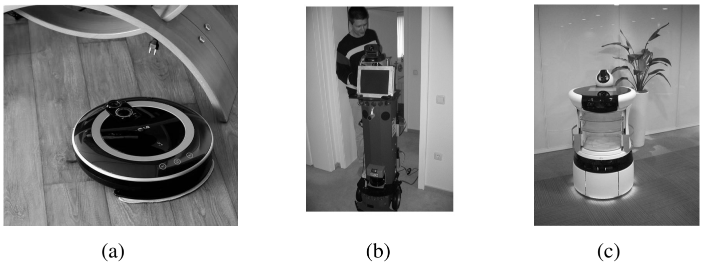


Figure 1.1: Example applications of vision-based mobile robots: (a) vacuum cleaner, (b) robot companion and (c) office delivery.

For these new types of applications it is obviously beneficial if the robot has a better means of sensing its environment. This can be achieved by mounting a camera on the robot which takes images while driving through the environment. These images can be used to build and maintain a representation of the spatial layout of the environment which can help the robot perform its tasks.

In this thesis, we will develop efficient and robust methods that allow a mobile robot to build and use such a spatial representation of a human inhabited home environment with camera images.

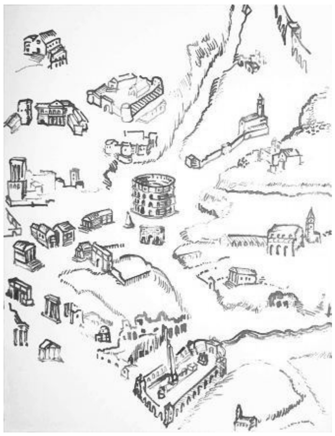


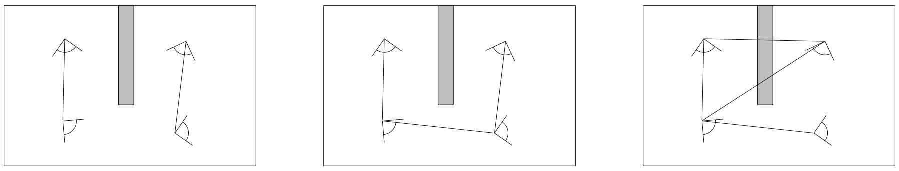
Figure 1.2: Pictorial map of Rome from the Middle Ages.

1.1 Representations of space

The spatial layout of an environment can be represented in different ways. Perhaps the most intuitive representation is a two dimensional geometric map, in which spatial structures such as walls and objects are represented by sets of 2D coordinates in a Cartesian coordinate system (Thrun, 2002). A variant of this geometric map is the 3D map, or 3D reconstruction (Pollefeys et al., 2008). A popular type of 3D geometric map in the field of robotics is the landmark-based map, which contains the 3D position of salient points in the environment. Examples of such salient points are sharp corners of furniture and bright spots on a wall.

Besides geometric maps, there are other types of representations of the environment that we will also denote with the term *map*. Some maps incorporate information describing how different parts in the environment look. In Figure 1.2, a so-called pictorial map of Rome is shown as it was seen through medieval eyes. Each small drawing represents one of the tourist sites. Typical for such a map is that the placement of the drawings in the map is not that important, as long as the viewer can recognize the sites and see which sites are close to each other.

Mobile robots equipped with a camera commonly use comparable representations of the environment (Kröse et al., 2001). A map can be composed of a set of images taken in the environment combined with a graph representation that links pairs of images. A link between two images indicates some spatial relationship between the two of them. Usually, quantitative information on this specific spatial relationship is stored with each link. In general, we call such a map a *view-based map* (Mallot and Gillner, 2000; Eustice et al., 2006). It is quite easy to use such a view-based map for robot localization. While driving around in the environment, the robot can take an image using its camera and, by comparing the image with the images in the view-based map, it can deduce where it is in the map. Besides this straightforward usage of view-based maps, they also scale well to larger environments (Bailey and Durrant-Whyte, 2006; Cummins and Newman, 2009).



(a) Similar semantic place (b) Reachable through motor commands (c) Relative pose can be estimated

Figure 1.3: A toy example showing the subtle differences between different types of spatial relationships. The figure shows two rooms seen from above. The four small eye shaped elements indicate the different poses of the robot from which images were taken. Lines between the poses indicate that there is a link between the corresponding images in the view-based map.

1.2 View-based maps

Each link in a view-based map indicates some spatial relationship between two images. Different definitions for the type of spatial relationship exist in the field of view-based maps and each results in a different map (see Figure 1.3).

For some maps, a link between two images means that they were taken at the same semantic place, such as “Kitchen” or “Library”, see Figure 1.3(a). This type of relation is interesting when doing semantic place recognition used for interacting with humans accompanying the robot (Topp and Christensen, 2010; Vasudevan and Siegwart, 2008; Torralba et al., 2003). It is not useful for other robot tasks, such as planning paths through the environment or goal directed navigation, because it lacks links between images taken at different semantic places.

For other maps, links indicate that the pose from which an image was taken can be directly reached from the place where another image was taken by sending some simple motor commands to the robot (see Figure 1.3(b)). This definition is commonly used in early literature on topological robot mapping, which combines the captured images with the low level motor control (Kuipers, 1978). The resulting map thus models a fixed set of paths through the environment which can be traversed by sending the corresponding motor commands.

A third type of link definition is based solely on the images themselves (see Figure 1.3(c)). A link between two images is defined as *the ability to estimate the relative camera pose up to scale given the two images*. This pose estimate is then stored with the link. A relative pose can be estimated from the projection of static structures on the two images, such as textured walls and objects in the environment. We emphasize that the scale of the relative camera pose can not be deduced from two camera images, unless we would make specific assumptions about the environment, such as the average distance to objects. Nevertheless, in this thesis we will commonly omit writing “up to scale” and just write “camera pose” or “pose”. A view-based map, based on this link definition, can be used for different robotic tasks. However, the set of images must be sufficiently dense. The estimated camera poses can be used for vision-based robot navigation because it allows to determine the driving directions to drive over arbitrary paths of linked images. However, keep in mind that it should be combined with a suitable obstacle avoidance system. Moreover, the relative poses can form the basis for a geometric map as follows. As a first step, they are used to estimate the camera poses in a single Cartesian frame. Then, structures as seen in the images are reconstructed in this frame by using triangulation.

1.3 View-based map building

For the autonomous vacuum cleaning application we cannot assume that a map, such as a view-based one, is available when the robot is deployed in its environment. This is true for a lot of the robot applications that appeared in recent years. Thus, the robot has to build such a map itself from images it took while exploring its environment. This important robotic task is known as mapping. The core of building a view-based map is to compare pairs of images. If an image pair is spatially related according to the used link definition, a link is added to the map.

For most applications, the robot needs to use its map while it is exploring the environment. For example, it might need to ask questions to a human guide to resolve certain ambiguities during the mapping process. To achieve this, the robot has to incrementally build a map by adding newly taken images to it and add links from each new image to other images in the map. Of course, the number of possible links that should be added increases with the number of images in the map. The challenge is to develop a view-based mapping method that is efficient enough to be run in “real time” yet robust enough to cope with the difficult circumstances of dynamic home environments.

The mapping problem is commonly approached by incrementally building a *landmark-based geometric* map. This geometric map makes it possible to predict where landmarks will be projected on newly taken images. These predictions can then be used to improve the efficiency, as well as the robustness of the mapping process. The approach of incrementally building a geometric map is known as SLAM, which stands for Simultaneous Localization And Mapping. Research in SLAM has focussed on efficiently and consistently integrating landmark observations and estimated relative camera poses, and has resulted in various well founded methods (Bailey and Durrant-Whyte, 2006). These have been successfully applied to build 3D landmark-based geometric maps of small scale office environments (Davison et al., 2007) and outdoor environments (Clemente et al., 2007). So why are we interested in view-based representations?

The landmark-based SLAM approach tries to build a representation of the environment that is independent of the actual images in which the landmarks were observed. Of course there is a dependency between landmarks observed in the same image. SLAM methods try to take this into account by modeling these dependencies. Unfortunately, such a model is not perfect and some dependency between the landmarks is lost. This is the case, for example, in environments with relatively big changes in lighting conditions as they typically affect the complete image.

View-based methods avoid this problem by explicitly maintaining a set of features, such as observed landmarks, separately for each image. The map can be used for easy, view-based, localization and visual navigation. It can also be used to build a geometric representation and can therefore be seen as a lazy approach, in which a relatively high amount of sensor information is retained.

The question we address in this thesis is how to deal with the effort of comparing new images with the map, since the view-based map cannot benefit from the predictions of landmark observations.

1.4 Objective of this thesis

In this thesis we focus on view-based mapping methods for real home environments. The aim is to develop a practical mapping method that is both robust in these challenging environments and efficient enough to be applied in real time. We focus especially on developing a robust and efficient image similarity measure, which is based on the ability to estimate the relative pose given two images. For unrestricted robot motion, one can take advantage of the well-studied methods originating from computer vision, see for example Hartley and Zisserman (2003). We will try to improve on these methods by making extensive use of the fact that the mobile robot drives over an approximately planar surface and the camera is mounted rigidly onto the robot. We have formulated two research questions:

- How can the planar motion assumption be used to estimate the relative pose given two images taken in a home environment?
- Can we thus obtain more accurate and efficient results than existing unrestricted pose estimation techniques?

In order to determine whether a link should be added to the view-based map, we have to assess the quality of the estimate. We will propose a new quality measure that is founded on probability theory. In particular, we will try to determine a probability distribution over all possible relative poses, given the two images, and use that to measure the uncertainty of the pose estimate. This uncertainty could directly be used to aid 3D reconstruction applications. We try to answer the following questions:

- How can the uncertainty of the estimated relative pose be measured?
- Does this uncertainty measure provide an efficient and robust image similarity measure for the view-based mapping application?

Both the pose estimation and the uncertainty estimation methods will be designed such that they need only little computation time and outperform existing methods on efficiency as well as accuracy. However, when adding images to a large view-based map, determining the similarity with all images is impractical. Instead of trying to make the similarity measure even more efficient and risking loss of accuracy, we compare each new image to only a subset of images. This subset of images should of course give a good representation of the complete set of images in the map. Relevant questions are:

- How can we limit the number of image comparisons necessary for view-based mapping?
- What effect does comparing with only a subset of images have on the resulting map?

Answering these questions allows us to build efficient and robust algorithms that can be implemented on robotic systems performing applications in real time. In this thesis, we will not focus on the robotic systems themselves. Nevertheless, in Appendix B we describe - as a proof of concept - a robot that uses some of the proposed methods for path planning and navigation among humans. In Appendix C, we show a similar robot system which maps a home environment in real time while resolving ambiguities by interacting with a human guide.

1.5 Thesis overview

Chapter 2: View-based mapping, an overview

In the field of robotics there has been, and still is, a lot of interest in view-based mapping. Here, we give a more elaborate description of the problem of building a view-based map and relate it explicitly to the problem of comparing images. An overview is given of the different approaches to compare images and how they are used in view-based mapping methods. In addition, we show how topological and possibly geometric information is used to improve the robustness and efficiency of map building.

Chapter 3: Two-view pose estimation assuming planar camera motion

In this chapter, we focus on comparing two images from the same scene and estimating the relative camera poses up to scale. We thoroughly investigate how the assumption, that the robot moves over an approximately planar surface, can be used to estimate the relative camera pose from a set of automatically extracted point correspondences. A new method is developed that estimates the full likelihood of all possible poses. The resulting maximum likelihood solutions are shown to be more accurate than state of the art methods.

Chapter 4: Image similarity for view-based mapping

Based on the developed pose estimation method, we propose a new image similarity measure for finding links for a view-based map. The method uses the full likelihood to directly approximate the probability that the estimated pose is the correct one. This approximation is evaluated using ground truth pose information. On top of this, we compare the resulting image similarity measure in a semantic place recognition task.

Chapter 5: Data association using connected dominating sets

Practical view-based mapping methods should be able to deal with large maps consisting of more than a thousand images. In this chapter we use a graph theoretic method called the Connected Dominating Set to obtain a minimal set of images that still adequately represents the complete map. It is compared with other frequently used methods such as subsampling over time or over the distance traveled.

Chapter 6: Conclusions

In this final chapter we summarize our main contributions and experimental results. In addition, we give directions for future work, focussing on promising applications.

Appendix A: Real home datasets

Throughout this thesis, we use the same datasets to evaluate each of the developed methods. These datasets were taken by two mobile robots in three real home environments with people walking near the robot and typical bad lighting conditions, resulting in challenging images.

Appendix B: Navigation using an appearance-based topological map

To show the usability of a view-based map, we describe a system that performs basic path planning and goal directed navigation using a map that was built with methods described in this thesis.

Appendix C:

In some applications, it is essential that a robot can map a new environment in real time. A good example is guided mapping, in which the robot needs to interact with a human guide to obtain semantical information and to resolve ambiguities while mapping. In this appendix, such a system is described.

2 Mapping based on images: an overview

This chapter gives an overview of the field of view-based mapping and describes a general framework for a view-based mapping system used in the rest of this thesis. As we will see, most view-based mapping systems directly or indirectly base the existence of a link between two images on some form of relative pose estimation. In addition, most mapping systems use topological and robot pose information while mapping.

2.1 Introduction

The problem of view-based incremental mapping, or simply *mapping*, can broadly be defined as trying to build and maintain a spatial representation, or *map*, of the environment while gathering sensor input including new images (Thrun, 2002; Bailey and Durrant-Whyte, 2006; Newman and Ho, 2005). The quality of such a view-based map is partly defined by its intended use. This could be goal-directed navigation, measuring distances, human robot interaction or - as often is the case - a combination of the three. Nevertheless, we can specify two requirements that are universal.

In the first place, the computational cost should be such that the procedure can run in real time on the robot while it is driving through the environment. Because modern robots and computers have a lot of memory and storage space, the main bottleneck is usually the required computation time.

In the second place, mapping methods should build *consistent* maps. It is difficult to properly define this second requirement. The idea is that images taken at very different places in the environment should not be close together, or linked, in the map. Vice versa, images shot from more or less the same position viewing more or less the same structures of the environment should be linked or placed close to each other. Anati and Daniilidis (2009) denoted these two concepts "spatial distinctiveness" and "compactness".

During the mapping process, the robot sequentially collects images. Each time the robot takes a new image, it should try to improve the map. The task of determining to which parts of the map a new image belongs, is known as *data association* (Bailey and Durrant-Whyte, 2006). For view-based mapping approaches, data association involves comparing the new image with previously acquired images that are already in the map. In Section 2.2, we give an overview of the different methods commonly used for comparing pairs of images.

What makes the mapping application interesting is the fact that a previously visited part of the environment can be revisited. If this is detected by the mapping algorithm, then it is called a *loop closure*. These detections are essential for a useful map, because they allow to plan new routes through the environment, which are distinct from the route already traversed by the robot.

It could also be that two different, spatially distinct, parts of the environment look alike, possibly resulting in a false loop closure. In view-based mapping, such phenomena are known as *visual aliasing* or *image aliasing*. Visual aliasing is usually catastrophic for the map, because it creates false links in the map. These links are sometimes called "worm holes" (Olson, 2008) and seem very useful for traveling directly from one end of the environment to the other. There is clearly a trade off between finding all loop closures and avoiding visual aliasing.

To limit the computational resources and avoid visual aliasing, mapping systems can take advantage of the sequential characteristic of the incoming sensor data and the information contained in the growing map . This can, for example, be achieved by only comparing to a subset of the images, called *key images*. There are methods that do this, using only topological information of the map built so far. This is described in Section 2.3. Mapping methods that also estimate and maintain the geometric robot trajectory in the map, have the ability to use relative pose information to help solve the data association problem. This is described in Section 2.4.

2.2 Methods for comparing images

Data association is a fundamental task in view-based mapping and in its core is usually an image similarity function that compares images. Given two images I_i and I_j, its aim is to determine how similar they are. This is expressed by some similarity value $S_{ij} = S(I_i, I_j)$. Commonly, this similarity value is derived from a dissimilarity value D_{ij}. In this chapter, we sometimes assume that this dissimilarity value is also available.

The common assumption is that if two images look similar, then there is a high chance that they partly depict the same structures of the environment. Thus, they should be linked in the map. The question is how to define similarity in images and how to measure it. Various answers are found in the field of computer vision or, more specifically, in the field of *content-based image retrieval* (Datta et al., 2008; Smeulders et al., 2000). In content-based image retrieval, the problem of finding images that depict the same part of the environment is called *place recognition* (Torralba et al., 2003; Konolige and Bowman, 2009). As we have mentioned in Chapter 1, the aim of place recognition is to find images that have the same semantic label. Take, for instance "Kitchen".

For maps aimed at goal-directed navigation or 3D reconstruction, we want a more specific requirement for the image similarity measure. Namely, two images are similar if we can determine some local geometrical information relating them. In the following part, we describe the problems faced when defining an image similarity measure and we give an overview of the different (dis)similarity measures used in existing view-based localization and view-based mapping methods.

2.2.1 Invariance vs specificity

Images can contain a lot of information about the environment. However, the raw pixel values of two images depicting the same structures of the environment are usually very different. On the one hand, this difference is caused by changing conditions of the environment, such as a changing illumination, displaced furniture and dynamic objects and people. On the other hand, images are influenced by changes of the measurement system itself, such as noise of the image sensor, a changing auto gain and different camera pose.

A similarity measure should be invariant or mostly invariant to these changing conditions, while being specific for the structures of the environment that are depicted in the images. Methods to reach such invariance can be roughly divided into two approaches.

A straightforward approach is to determine from each image a set of *image features* which are invariant to each condition. For example, to reach invariance for the changing color of the light source one can use only the light intensity of each pixel and discard the chrominance by converting color images to gray scale (Lowe, 1999).

A more involved approach to reach invariance to a certain condition is to explicitly measure its value from each image and take it into account when comparing images. An example is described in Pollefeys et al. (2008), where the gain factor of the camera is estimated for each of the images. When comparing two images, the features are normalized using the specific estimated gain factors.

As we shall see in the next sections, most image comparison methods used in existing view-based mapping systems use the straightforward approach for most changing conditions. They determine a set of invariant image features for each image. However, this is not the case for the changing camera pose. The relative camera pose between two images is usually explicitly estimated, based on the extracted image features. When determining the image similarity, this estimated pose is used to compensate for the variation in the feature sets that resulted from the change in camera pose. This special treatment of the camera pose, reveals the implicit assumption that for view-based mapping most of the variation in the images can be explained by the difference in camera pose.

2.2.2 Image features

Most, if not all, image comparison methods first try to determine some quantitative *features* that are specific for the structures - such as textured walls and objects - as depicted in the images and invariant or mostly invariant to other influences. The set of feature values extracted from an image is known as the *feature vector* or *descriptor*. Image comparison methods can be roughly divided into two groups, depending on the type of features they use (Murphy et al., 2006). The first approach describes the each image as a whole, using a fixed length descriptor which is called a *global feature*. The global feature is usually constructed such that the Euclidean distance between the descriptors d_i and d_j of two images i and j defines the image dissimilarity measure:

$$D_{ij} = \|d_i - d_j\|, \tag{2.1}$$

The second approach first searches for salient regions in the image and describes each region with a separate descriptor (Harris and Stephens, 1988). A complete image is then composed of a set of *local feature* descriptors. The size of this set can be different for each image. Determining the similarity is then based on finding feature matches, which can be accomplished in various ways. In the following sections we give examples of commonly used global and local features.

Global image features

Describing images with global image features and matching pairs of global descriptors are usually relatively simple procedures compared to using a local features approach. For example, an image can be described by its average pixel intensity or higher order moments and Eq.(2.1) can be used to compare these values. Although such a method is very crude, it costs very little computation time. In Gross and Koenig (2004) and Murillo et al. (2007) it is used in relatively large view-based mapping applications as a first check to decide on which image one should apply more sophisticated time consuming methods. Another simple and efficient method is using color histograms. A well-known study described in Ulrich and Nourbakhsh (2000) applies them for both indoor and outdoor view-based localization.

A somewhat more involving approach, is the use of dimension reduction techniques like Principal Component Analysis (PCA). This is popular for panoramic images. By treating the raw pixel values of a set of training images as vectors, one can apply PCA to obtain a number of eigenvectors corresponding to high eigenvalues. These eigenvectors are known as eigenimages and describe the main variance in the image set. Each newly acquired image is then projected onto these eigenimages, resulting in a concise descriptor vector. Examples of robot localization methods using such a dimension reduction scheme are described in Nayar et al. (1995); Kröse et al. (2001); Jogan and Leonardis (2003). They are usually combined with methods that estimate the relative rotation of the camera, as explained in Section 2.2.3. An important characteristic of this approach is that a training set of images is used to determine the eigenimages. This is problematic when mapping an unknown environment, but could also be an advantage if a representative set of images is available. A related approach uses the magnitude coefficients of the Fourier transform as the global feature descriptor instead of the eigenvalues (Ishiguro and Tsuji, 1996; Menegatti et al., 2004). The advantage of this approach is that, if the robot drove over a planar surface and took omnidirectional images using a fisheye lens or a convex mirror, then the descriptor is invariant to the robot rotation.

The main drawback of global features is that they lack invariance against viewpoint changes. Indeed, most of the methods cited above are applied in relatively simple indoor office environments. Because mapping methods are nowadays applied in more taxing environments and more computational power is available, one might think that they are outdated. However, when they are combined with other sophisticated mapping techniques, they are still very usable. Milford and Wyeth (2008) describe a localization and mapping system, which is able to build a large outdoor map while using a simple global feature method to compare images. In this case, the images were described by the sum of pixel intensities of each image row.

Local image features

Usually, some regions of an image have more visual texture than others. One can take advantage of this by finding highly textured regions and describing them separately. Each of these regions is characterized by a certain image point and described using the surrounding pixels. Together these are called a *local feature* or simply *feature*. Comparing two images then entails finding similar local features between the images. This type of image comparison is popular, because it is more robust against occlusions and changes in lighting than methods based on global features. Building an image descriptor involves two tasks. First, salient features are detected. Second, feature descriptors have to be built.

There are various point image feature detectors to extract a set of salient features from images, of which we name the most popular in the field of view-based mapping. A computationally cheap method is to extract Harris corners (Harris and Stephens, 1988). These are, however, not invariant for illumination changes and are mainly used in small scale landmark-based SLAM approaches (Davison et al., 2007) relying highly on the estimated landmark positions in the map, or in combination with other more robust features (Eustice, 2005). Popular robust features are blob-like points such as used in SIFT (Scale Invariant Feature Transform) (Lowe, 1999) and SURF (Speeded Up Robust Features) (Bay et al., 2006), which were originally meant for object recognition tasks and are relatively time consuming. At the time of this writing the majority of view-based mapping methods uses one of these two features. Nevertheless, a more appealing approach to finding viewpoint invariant features is through *tracking* local features over a sequence of images. Points that can be tracked for a certain number of images are bound to be good for comparing images. There is a lot of literature on tracking simple features such as the famous KLT tracker (Shi and Tomasi, 1994). However, more sophisticated feature trackers, such as SURF trackers are, still under development (He et al., 2009).

In order to build descriptors for the extracted feature points, various methods exist. A simplistic approach is to use the pixel intensities surrounding the salient feature. These can be compared using the Euclidean distance, known in this context as SSD (Sum of Squared Differences) or the L1-norm, known as SAD (Sum of Absolute Differences). Such simple methods are sometimes used in Computer Vision applications (Hartley and Zisserman, 2003). However, they are not popular in the field of robotics because they lack robustness against illumination changes. More involved descriptors are based on particular feature types and even share some of the needed processing steps with the detectors. This is the case for SIFT and SURF features. When a study claims to use SIFT features, then both the SIFT detector and the SIFT descriptor is used, unless otherwise stated. The same holds for SURF features.

The local feature descriptors of different images are compared to find matching features. Like global feature descriptors, local feature descriptors are usually designed to be compared using the Euclidean norm. To find corresponding features between two images, the distances between all features from one image to the features in the other image should be found. Two features can then match, if the distance between their descriptor is smaller than a certain threshold. However, it is more common to correspond each feature in one image to the nearest neighboring feature in the other image, if the ratio between the distance of this nearest neighbor and the distance to its second nearest neighbor is above some threshold, typically set to .8 (Lowe, 1999). In this way, one ensures that the correspondence is more or less unique.

The image similarity measure is usually based on the percentage of image features that were matched, as for example set out in Andreasson et al. (2008):

$$S_{ij} = \frac{M_{ij}}{\frac{1}{2}(F_i + F_j)}, \tag{2.2}$$

where M_{ij} denotes the number of matches and F_i the number of features found in image I_i. However, other schemes exist ranging from very simple functions such as $S_{ij} = M_{ij}$

in Valgren and Lilienthal (2007) to complicated functions such as in A. C. Murillo and Sagues (2008):

$$S_{ij} = e^{-\frac{1-s_{ij}}{\sigma_S}} \text{ with } s_{ij} = \frac{1}{M_{ij}d_e + zF + 1}, \tag{2.3}$$

where d_e denotes the average Euclidean distance of the feature matches, F the number of unmatched features, z a constant penalty term and σ_S the variation of s_{ij} for different images I_j.

An important drawback of comparing local image features is the computation time used. Typically, one image results in about 10^3 local features each with a descriptor of typically 128 byte values. Comparing two images thus results in the order of 10^8 operations. A possible solution is to use Approximate Nearest Neighbor algorithms (Arya et al., 1998) as done by Sim et al. (2005) for landmark-based SLAM and A. C. Murillo and Sagues (2008) for view-based mapping.

Recently, it has become popular to perform dimension reduction on the descriptors by quantifying them using a so-called code book. The code book contains a mapping from descriptor space to a limited number of discrete values called words. Such an approach is known as the Bag of Words (BoW) approach and originated from the field of natural language processing (NLP). A set of representative training data is used to construct the code book. Image comparison is then performed by comparing the specific word counts of each image. With this efficient comparison technique, maps can be constructed in the order of 10^3 images. This is done by Fraundorfer et al. (2007); Konolige and Bowman (2009); Callmer et al. (2008); Schindler et al. (2007). When using a BoW approach, one can benefit from the availability of various techniques studied in NLP. An example is dealing with the interdependencies between different words, used in the popular view-based mapping framework “FABMAP” (Cummins and Newman, 2008, 2009). FABMAP can build maps of 10^5 images.

In addition, there are localization and mapping systems that use line features from images. This mostly concerns vertical lines, which project to vertical lines in the image if the robot drives over a planar surface. They are, for example, used in the Fingerprint method (Lamon et al., 2001) in combination with color information and in Scaramuzza et al. (2009), where a line descriptor is developed that is similar to the SIFT-descriptor for point features.

2.2.3 Using local geometric information

As we have mentioned in Section 2.2.1, most view-based mapping systems estimate the relative camera pose from the images in order to determine the image similarity. In this section we give an overview of the specific methods used in existing mapping systems. The general approach can be described as follows. First an image similarity measure is defined on the basis of the existing feature matches as found using one of the discussed feature extraction methods. The resulting similarity usually depends greatly on the particular poses of the camera. The image or the extracted features can be shifted or changed to compensate for different relative camera poses. This is performed for a subset of possible poses and the image similarity value is determined. The highest value is then taken as the final image similarity:

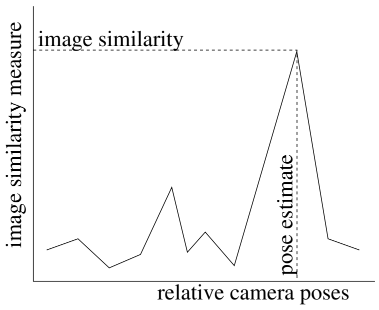


Figure 2.1: Abstract graph clarifying the relation between the estimated relative camera pose and the determined image similarity. The horizontal axis represent all possible relative poses and the vertical axis the image similarity measure $S\left(I_i, Tr(I_j)\right)$ (see Eq. 2.4). The maximum of the function indicates both the best fit relative pose and a measure of the image similarity.

$$S_{ij} = \max_{Tr} S\left(I_i, Tr(I_j)\right), \tag{2.4}$$

where $Tr(I_j)$ denotes the changed image I_j given the pose Tr. The idea is that the pose Tr that maximizes the function corresponds to the actual relative camera pose, thus removing all the variance in the images that was due to the different camera pose:

$$\widehat{Tr}_{ij} = \arg\max_{Tr} S\left(I_i, Tr(I_j)\right). \tag{2.5}$$

See Figure 2.1 for a graphical representation of the relation between the image similarity measure and the relative pose.

An important consequence is that the resulting image similarity measure reflects how well local geometrical information can be extracted from the images. This is useful for goal-directed navigation or global geometric map building, as introduced in Chapter 1. In the following part, we discuss the specific techniques that are used for both global and local image features.

Geometric information for global image features

Global feature comparison techniques are mainly used for robots that move over a planar surface, which makes it easier to estimate the relative pose. In addition, most methods make the implicit assumption that the shift in appearance can be mostly explained by a camera rotation around the vertical axis. Estimating this rotation requires finding the horizontal shift of the images that maximizes the similarity between them.

For PCA-based image comparison, this can be readily performed by shifting the eigenimages horizontally, for example done by Jogan and Leonardis (2003). For rotation invariant features, such as color moments, the image can be horizontally segmented into

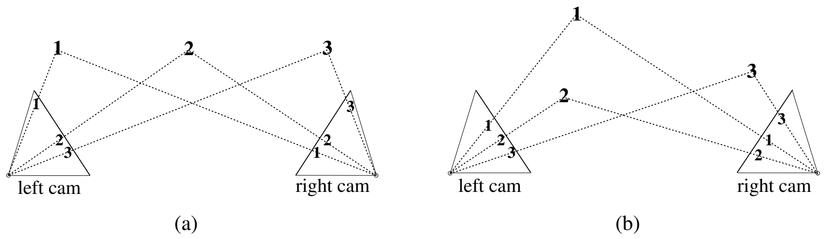


Figure 2.2: Top view of three numbered objects that are projected on the image planes of two cameras. (a) As can be seen for different camera positions the relative position of the projected objects is not preserved, that is: the left camera sees object 2 and 3 relatively close to each other, but the right camera sees 1 and 2 closer. For this object configuration, however, the order is the same for both viewpoints. (b) For general object configurations neither the relative position nor the order is preserved: the left camera sees 1-2-3, while the right camera sees 2-1-3.

vertical rows, which are described separately. Rotation estimation is performed by shifting these rows. Examples of robot localization methods that use such an approach are described in Gross and Koenig (2004); Sturm and Visser (2009). The RatSLAM system (Milford and Wyeth, 2008) uses a similar approach in combination with image intensity information per pixel row and shifting these pixel rows to get the highest similarity.

Geometric information for local image features

Some local feature methods use the same assumption for the relative pose as used by the global feature methods, in which a 2D rotation is estimated by determining the horizontal shift of local feature correspondences. This is for example used in the view-based mapping systems FABMAP 2.0 (Cummins and Newman, 2009) and MiniSLAM (Andreasson et al., 2008). This assumption does not hold if the camera also translated, as shown in Figure 2.2(a).

In Lamon et al. (2001) a different type of feature consistency is used. Instead of assuming only 2D rotation, it is assumed that the robot is moving in a convex space, e.g. an empty room. Such an assumption does not preserve the relative angles between observed features, but does preserve their order, as can be seen in Figure 2.2(a). This approach is commonly used in combination with vertical line features such as described in Murillo et al. (2007). It is also common in robot homing applications that try to perform goal-directed navigation given an image taken at the goal position (Franz et al., 1998; Argyros et al., 2005). In general, for non convex spaces, this assumption does not hold. This is shown in Figure 2.2(b).

Only a few methods were proposed that do take both rotation and translation into account, while still enforcing motion over a planar surface (Ortín and Montiel, 2001; Goedemé et al., 2005a). In Scaramuzza et al. (2010) a method is presented that, in addition to the planar motion constraint, uses automobile vehicle constraints to improve motion estimation.

Most localization and mapping systems do not make assumptions about the structure of the environment or particular constraints of the robot motion. Both 3D rotation and 3D translation up to scale are estimated from the feature correspondences. This estimation can be performed using various methods developed in the field of Computer Vision (Kanatani, 1996; Torr and Murray, 1997; Hartley and Zisserman, 2003; Schaffalitzky and Zisserman, 2002).

The most popular estimation method at the time of this writing combines standard least squares schemes with the robust RANSAC algorithm (RANdom SAmple Consensus) (Fischler and Bolles, 1981). This method directly estimates both the relative pose and the number of correspondences that fit the relative pose. It is used in various recently proposed mapping systems (Eustice, 2005; Newman et al., 2006; Fraundorfer et al., 2007; Konolige et al., 2009; Segvic et al., 2009). The same scheme is also popular in the image retrieval community for the application of place recognition under the term "spatial verification" (Chum and Matas, 2010; Li et al., 2008; Philbin et al., 2007).

2.2.4 Discussion

We described relevant image comparison methods used in view-based mapping. Although this overview is far from exhaustive, it is clear that there are a lot of different choices. Unfortunately none of the these can be regarded as the "golden standard".

The most popular approach is to use SIFT and SURF features in combination with the Bag of Words approach and a local geometric consistency check. This consistency check is actually not integrated in the BoW approach, but used more as an ad hoc rule to find image pairs that wrongly got a high image similarity value. A better integration of the geometric consistency and appearance similarity would seem to be a logical improvement.

The geometric consistency check involves estimating the relative camera pose from a set of feature matches. If this estimation fails, this will result in a low image similarity value. Thus, most of the used image similarity measures actually try to measure how well a relative pose can be estimated from two images. This is exactly the definition of a link between two nodes in a view-based map, as proposed in Chapter 1 and used throughout this thesis.

2.3 Exploiting topological information

As introduced in Chapter 1, a view-based map can be seen as a graph $G = (V, E)$ in which a node $v \in V$ represents an image and a link $(u, v) \in E$ denotes that the two images corresponding to nodes u and v partly depict the same structures of the environment. It is straightforward to build such a map, using one of the discussed image similarity measures. For each new image the robot takes, the similarity is determined with every image in the map. If this similarity is above a certain preset threshold, then a link is added to the map. Commonly, the similarity value S_{ij} and possibly some local geometric information is then stored for each link.

Numerous view-based mapping methods use this straightforward approach (Newman and Ho, 2005; Ulrich and Nourbakhsh, 2000; Fraundorfer et al., 2007; Konolige and Bowman, 2009; Callmer et al., 2008; Schindler et al., 2007). However, this approach leans heavily on the assumption that if two images look similar, then there is a high chance that they partly depict the same structures. Robustness against visual aliasing is completely ruled by the robustness of the similarity measure. On top of this, it is computationally costly because each new image has to be compared with all images in the map.

The robustness and efficiency can be improved by taking specific properties of the mapping problem into account. In the following sections, we discuss the two main properties and how they can be used to improve data association. Images are acquired sequentially and previously visited parts of the environment can be revisited.

2.3.1 Image sequences

Images are acquired sequentially while the robot is driving around. This sequential property is sometimes directly used to improve mapping by always adding links between consecutive nodes (Valgren et al., 2007; Konolige and Bowman, 2009). These links are sometimes called “weak links”, as opposed to the links that are found by comparing images which are called “strong links” (Lu and Milios, 1997). The sequential property also results in a dependency of the features extracted from images taken close to each other in time. If one uses a tracker to identify salient image points, such as the KLT tracker, this dependency information is used on the image descriptor level. This is, for example, the case in Eustice (2005). More interesting is to use it on the data association level and improve both its robustness and speed.

Most large scale mapping methods do this by implicitly reducing the number of images in the map. This is done by processing only a limited number of images per time step, like in Ulrich and Nourbakhsh (2000), which uses only one image per second. One could see this as sampling the images uniformly over time. A similar approach is to process only a limited number of images per traversed distance, which requires a method for estimating the displacement. In Sim and Dudek (1999); Jogan and Leonardis (2003) this is done on the basis of odometry measurements and in Cummins and Newman (2008) on the basis of GPS readings. Both these approaches have their drawbacks, which are partly due to the dependency of the consecutive images on the speed of the robot. When the robot is standing still, captured images usually look similar. A time sampling-based method puts these almost identical images in the map, which could make data association unnecessarily slow. A displacement-based method would put only one of these in the map, not capturing possible dynamic changes in the environment such as illumination changes.

A better approach is to use the image similarity measure to define when a new image should be added to the map. For example, by only adding an image if the similarity with the previously added image drops below some threshold as is done in Konolige and Bowman (2009). In Kosecká et al. (2005) a more sophisticated approach is set out, in which a clustering algorithm is used to find groups of consecutive images that look alike, based on image similarity, and sample one image per cluster for data association.

Another straightforward method to use the sequential property, is to only define a link between two nodes if not only their images are similar, but also the images taken right before and after have a high similarity value. This is known as *smoothing* the map and is commonly used, for example as in Cummins and Newman (2008).

All these methods are relatively easy to implement and at the same time improve data association considerably. Most of them can also be used in other applications where images are shot in a sequence, for example when tracking moving objects in video. Other characteristics, more specific for the localization and mapping application, can also be used. This will be discussed in the following section.

2.3.2 Loop closing

While a robot is mapping, it usually observes at least some parts of the environment multiple times. This does not only occur during a loop closing event, but for example also when the robot is driving back and forth through a corridor. There will be a dependency between the features extracted from images observing the same part of the environment, although taken at different visits. Like for the sequentially shot images, this can be used to make data association faster and more robust.

A commonly used approach groups the set of images of the view-based map into clusters of similar looking images. From each cluster a single key image is then chosen which is used for data association. This is similar to the method in Kosecká et al. (2005) discussed earlier, but is not restricted to clusters of sequentially taken images. A question is how to efficiently solve such a complex clustering problem. A common solution is to use the normalized graph cut algorithm (J.Shi and J.Malik, 2000) as applied in view-based mapping (Zivkovic et al., 2005) and in a view-based SLAM method (Rogers and Christensen, 2009). In Valgren et al. (2007) an incremental clustering algorithm is proposed that clusters the graph while it is growing and uses it for view-based mapping. In addition, the mapping systems FABMAP and RatSLAM groups images. In FABMAP (Cummins and Newman, 2008) these groups are called "locations", which consist of images that observe the same objects. In RatSLAM (Milford and Wyeth, 2008) these groups are represented by pose cells, which are inspired on the brain cells a rat uses for mapping and trained using an associative learning scheme.

In Konolige and Bowman (2009) the problem of revisiting the same places is studied more thoroughly in the context of *lifelong mapping*, in which a robot keeps on updating its map over a long period of time. Images are grouped using a graph clustering method. In addition, it uses geometrical information removing images that were taken in a vicinity of 50 centimeters of another image.

In Zhou et al. (2008) a method was proposed that only adds those images to the map that reduce the amount of uncertainty of the map. In this case, the uncertainty was defined in a geometrical mapping context, but this could also be generalized for non-geometric view-based mapping. A drawback of such a method is that once an image is added to the map, it cannot be replaced by an even more informative image. Vice versa, images that are not added, could potentially prove to be informative on the basis of future images.

2.3.3 Discussion

View-based mapping systems can simply be based on one of the many image similarity measures defined in Section 2.2. However, to improve robustness and efficiency, view-based mapping systems often use the dependency between sequentially shot images and images observing the same part of the environment. These types of improvement can directly use the topological information available in the view-based map.

The most popular method is to first cluster the images in groups based on a graph clustering algorithm, and then pick a single image per cluster for data association. However, graph clustering is not trivial. It would be worthwhile to investigate if this problem could also be solved in one step, directly determining a set of key images. This would avoid solving the, perhaps more difficult, clustering problem.

2.4 Exploiting robot pose information

In addition to topological information, various view-based mapping methods make use of robot pose estimates when performing data association. It is evident that the probability that two images depict the same structures of the environment is very dependent on the relative pose between the two camera poses from which the images were taken. If the view-based map contains the camera pose for each image in a single coordinate frame in addition to an estimate of the camera pose of a new image, then these relative poses can be estimated directly and used to decide which image pairs are more likely to depict the same structures. The problem is how to obtain these camera poses from the unknown environment.

A solution is to estimate these poses from the view-based map itself, possibly using the local geometric information found during image comparison. Each new link between two nodes in the topological map poses a non-linear geometric constraint for the set of robot poses. The complete set of links of a view-based map in this context is sometimes called a *constraint map* (Konolige, 2005). The problem is to find all the camera poses that best fit these constraints.

This “geometric” mapping problem has attracted a lot of attention from the robotics community in the last twenty years, which resulted in an immense amount of literature. In this section, we give an overview of this field. First, we briefly describe some methods that base a geometric map solely on the similarity between pairs of images, which is a type of *embedding*. Then, we move to the more common method that fuses local geometric information extracted from the image pairs into a global map, which is the approach known as view-based SLAM or trajectory SLAM. Finally we explain how the estimated robot poses are used to improve data association.

2.4.1 Embedding

If images are compared using a global feature approach as described in Section 2.2.2, then commonly no explicit geometric information is available or, as explained in Section 2.2.3, only rotation information is known. However, there is no translation information. Thus, it is not that common to estimate camera poses based on an image comparison function that uses only global features. Nevertheless, a dissimilarity function does provide a measure of distance that can be interpreted geometrically. It has been shown in Verbeek (2004) that by simply applying dimension reduction on a set of images picturing a rotating head one can find a 2 dimensional ordering of the amount of rotation. Each image was thus embedded in a low dimensional manifold. In a similar fashion, one can find an ordering of the amount of translation from images taken by a driving robot. Effectively one is embedding the images in a low dimensional coordinate system.

For example, in Ham et al. (2005); Yairi (2007) PCA-like methods are used to project the images on a 2D or 3D manifold which gives an indication of the 2D or 3D position of the robot pose. For small scale environments this is shown to improve position estimates obtained through odometry. In Menegatti et al. (2004) a spring model was used to find robot poses that best fit the image dissimilarities which were computed using the distance between the magnitude components of their Fourier transform.

The method used in the RatSLAM system (Milford and Wyeth, 2008) is also a type of embedding. Here a non-linear mapping between images and poses is learned using a type of recurrent neural network model given an image dissimilarity measure based on pixel intensities. It is shown that with such an approach it is possible to robustly approximate a very large 2D robot trajectory from images only and use it to improve data association while mapping.

2.4.2 SLAM

Another, much more popular approach to estimate camera poses is to use the relative poses as estimated up to scale when comparing image pairs using local features (see Section 2.2.3). While the map is growing, new links provide new relative poses which can be used to re-estimate all robot poses. The estimation problem can, however, not be solved in closed form because of the non-linearity of the relative pose constraints. This problem is known as view-based SLAM (Simultaneous Localization And Mapping) (Bailey and Durrant-Whyte, 2006; Eustice et al., 2006) It is closely related to the well-known landmark-based SLAM problem (Newman, 1999; Smith et al., 1990), where the position of landmarks needs to be determined given their projection on the images. Because of the popularity of the term SLAM, it has also been associated with purely topological view-based mapping (Milford and Wyeth, 2008; Newman and Ho, 2005; Cummins and Newman, 2008). In this section, however, SLAM is reserved for the problem of finding the geometric robot poses or landmark positions from local geometric constraints.

View-based SLAM and landmark-based SLAM can be solved using the same type of approaches. The traditional approach is based on an Extended Kalman Filter, which linearizes each new constraint using the latest map estimate and uses that to maintain a full covariance over all robot poses (Smith et al., 1990). The disadvantage of this method is the large computational load and memory usage which both grow quadratically with the number of poses. Eustice et al. (2006) has shown that by maintaining the inverse of the covariance matrix, called the information matrix, these can be reduced considerably. The disadvantage is that the uncertainty of the pose estimate is not available, which has a negative effect on data association as we shall see later (Frese, 2006a).

In recent years, there has been a shift towards non-linear algorithms which are not based on the Kalman Filter. This is partly driven by advances in the very much related

Structure from Motion (SfM) problem studied in Computer Vision (Triggs et al., 2000; Ni et al., 2007). The main difference with SLAM is that SfM is defined as a non-iterative problem. This resulted in non-iterative methods such as Smoothing And Mapping (Dellaert and Kaess, 2006) and Multi-level relaxation (Frese et al., 2005), that for each new image estimates all poses using all local geometric constraints. Later iterative variants of these algorithms have been proposed, such as in Kaess et al. (2008).

A second source of non-linear SLAM approaches originated from advances in the study of Graphical Models (Bishop, 2006). Viewing the constraint map as a Bayesian network model and approximating it using tree like structures, allowed for new iterative non-linear SLAM methods. Examples are the sophisticated TreeMap algorithm (Frese, 2006b) and the TORO system (Grisetti et al., 2009).

These modern approaches to solving SLAM have been applied in relatively large areas. However, they depend heavily on successful data association of the previously taken images. If some visual aliasing is present, the estimation process will often fail catastrophically (Bailey and Durrant-Whyte, 2006). The problem is that these SLAM approaches are not robust against grossly outlying local geometric constraints. Indeed the results of the RatSLAM system, which is based on a type of embedding and not the SLAM methods discussed here, are unprecedented.

2.4.3 Gating

The estimate of the poses determined by the embedding or the SLAM process can be used to improve data association. From the pose estimates, the relative pose and its uncertainty between a new image and an image in the map can be determined. This relative pose can, on the one hand, be used to determine whether the images are likely to depict the same structures and if it is worth comparing them. On the other hand, if the images are compared, one can check if the resulting relative pose fits the estimated map. This last procedure is known as *gating* (Neira and Tardós, 2001; Bailey and Durrant-Whyte, 2006) and has proved itself for mapping small environments such as indoor office rooms (Davison et al., 2007). This is not the case for mapping applications in general. The main problem is that the estimated uncertainty of the poses is usually too small. Most, if not all, SLAM algorithms are said to be “overconfident” or “optimistic”, despite the large amount of research in this area. If gating were to be strictly applied, then most hypothesized loop closing links would not be accepted. This is the reason why, even for mapping systems that aim at building geometric maps, it is common to resort to topological data association to close loops (Newman and Ho, 2005).

Local parts of the geometric map can be used as a measure of displacement of the robot, which can then be used to define a set of key images as described in Section 2.3.1. It is used in Konolige and Bowman (2009) to remove redundant views from the image set. Furthermore, in some applications it is possible to explore the environment in such a way that no large loops are present, making gating possible. This is the case in most underwater exploration scenarios, such as described in Eustice (2005).

2.4.4 Discussion

View-based mapping systems could in principle benefit from geometric camera pose information. The problem of estimating a consistent set of camera poses given local geometric information can be concisely stated, yet is difficult to solve. It therefore attracted a lot of research and resulted in a large collection of different methods. Still, most pose estimation methods are fragile and for most applications the estimates are not suitable for data association. There is still a lack of understanding on how to properly fuse the qualitative information coming from image-based data association methods to estimate robot poses.

2.5 Conclusion

This chapter gave an overview of existing view-based mapping literature. Unfortunately, almost all studies focus on developing new methods, without performing proper comparisons with other work. It is, therefore, difficult to say which method will be favorable for application in a home environment.

What can be concluded is that most of the image comparison methods combine SIFT or SURF feature matching with some sort of local geometric check and thus, often implicitly, measure how well a relative camera pose can be estimated from the images. For efficiency and robustness, they commonly assume that the robot drives over a planar surface. Also this assumption is usually made implicitly. Thus, even though there exist a lot of methods that use the planar motion assumption, there is a lack of understanding how to properly use the planar motion assumption to estimate the relative pose given two images. In the next chapter, we will adress this question. After answering this question we try to define a better founded image comparison method, that is explicitly based on the uncertainty of the pose estimator.

Data association is commonly improved by using geometric positional information of previous camera poses. However, we have seen that methods to estimate these camera poses are still fragile. This will especially be the case in challenging environments such as homes. In this thesis we will not use camera pose estimates to improve data association. Rather, we will focus on the problem of using topological information to improve view-based mapping. We explained that some systems solve this by applying solutions for the graph clustering problem. We will not follow that route, but will investigate if there is a more direct method to use topological information to improve view-based mapping.

3 Two-view pose estimation assuming planar camera motion

In this chapter, we focus on comparing two images taken by a robot observing the same scene from different positions and computing the relative pose between the two robot poses.[1] We assume that the robot is equipped with a rigidly mounted calibrated camera under a known pose. In order to improve the robustness and efficiency, we make the assumption that the robot moves over a planar surface. We propose a completely novel histogram-based method to determine a probability density function over the space of all relative poses. Being able to robustly and efficiently estimate the relative pose, is essential for the rest of the thesis. On the one hand, the existence of a relative pose is used to define a link between two nodes of a topological view-based map in Chapter 4. On the other hand, the estimated relative pose can be used for goal directed robot navigation, see Appendix B.

3.1 Introduction

In Chapter 2, we have seen that various topological and geometrical mapping approaches are based on the ability to compare a pair of images. State of the art methods estimate the relative pose between the two camera poses from which those images are taken. A common way to do this, is to automatically find similar looking image points between two images. Part of the resulting image point correspondences are the projections of salient 3D points in the environment, called *landmarks*. This is used to determine the relative camera pose, consisting of the translation and the rotation using closed form least squares methods (Hartley and Zisserman, 2003). Actually, the scale of the relative pose cannot be determined. It is lost during the projection step. In this chapter, however, we use the term “relative pose” instead of “relative pose up to an unknown scale”.

A major challenge when determining the relative pose given point correspondences, is that a large percentage of the similar looking image points do not correspond to the same 3D landmark. They are so-called *mismatches*. This percentage can increase up to 100% due to changing lighting conditions, large distances between the images and big viewpoint changes. In addition, the image point locations of correct matches are noisy. This noise stems from various sources, such as the noise of the imaging device itself, discretization, errors of the calibration and so on.

To cope with noise and mismatches, the closed form methods have to be combined with so-called *robust* algorithms. There are 3 robust methods commonly used: RANSAC (RANdom SAmple Consensus) (Fischler and Bolles, 1981), M-Estimators (Maximum

[1] Part of the work described in this chapter was presented at the Robotics: Science and Systems Conference (Booij et al., 2010).

likelihood Estimators) (Torr and Murray, 1997) and the Hough Transform (Hough, 1962). State of the art relative pose estimators combine RANSAC to estimate an initial relative pose and M-Estimators to improve it (Hartley and Zisserman, 2003; Eustice, 2005; Cummins and Newman, 2009; Nistér et al., 2004). Basically, this approach tries to find a maximum likelihood solution by first rejecting outliers, which are mismatches and correct matches with too much noise, using an error threshold. It then bases a least square solution on the remaining correspondences, the inliers. The success of this approach depends on the number level of noise and the percentage of mismatches (Torr and Murray, 1997; Zhang, 1998; Brückner et al., 2008; Armangué and Salvi, 2003). The Hough Transform on the other hand, if seen probabilistically, computes the full likelihood on a discrete grid of poses, without making an explicit distinction between inliers and outliers. Because computer memory requirements grow exponentially with the number of parameters, the Hough Transform is generally not suited for pose estimation problems. However combinations of the Hough Transform with RANSAC (den Hollander and Hanjalic, 2007) do exist, as well as methods that treat rotation and translation estimation separately (Heeger and Jepson, 1992; Censi and Carpin, 2009).

An approach to make pose estimation easier, is to incorporate constraints on the possible relative poses. If the robot drives over a planar surface, then the camera can only rotate around a certain fixed axis which is perpendicular to the two dimensional translation direction. This is not only the case for indoor wheeled robots, but also for outdoor vehicles driving over planar roads. We will call this constrained version of the general pose estimation problem the *planar relative pose problem*. Given that there are fewer possible relative poses for the planar relative pose problem, incorporating this constraint results in more robust and accurate estimation process. Because the problem has fewer parameters to estimate, the pose can in principle be estimated with fewer correct correspondences than the general unconstrained pose estimation.

The question is how to incorporate the planarity constraint in a principled manner when estimating the relative pose from point correspondences. What is the minimum number of required correct correspondences? And how does one deal with a larger number of correspondences which include a high percentage of mismatches?

In this chapter the application of the planar constraint for estimating the two-view relative pose from point correspondences is investigated thoroughly. We first describe the problem more formally in Section 3.2 and discuss previous work dealing with it. An important aspect is the number of degrees of freedom of the planar relative pose problem and the number of correspondences needed to solve it.

It is assumed by Ortín and Montiel (2001); Goedemé et al. (2005a) that the planar relative pose problem can be solved using two correctly matched point correspondences. However, in Section 3.3 we provide the novel insight that two correspondences sometimes could have been the result of two different relative camera poses. In other words, the problem can sometimes not be uniquely solved using two correspondences. For uniformly distributed landmarks, we show that this actually happens in 50% of the cases. We describe these cases on the basis of the position of the landmarks.

In Section 3.4, we derive the specific trigonometric functions that relate a single point correspondence to possible relative poses that fit this correspondence. This results in a novel closed form 2-point algorithm that given two point correspondences computes a solution, or, if it cannot be uniquely solved, both solutions. We call this method the

Planar Two Point algorithm.

We continue the chapter by investigating how to deal with correspondences that include a high percentage of mismatches. In Section 3.5, we briefly describe a state of the art approach to solving the planar relative pose problem. This is a variation of the standard approach to perform general unconstrained pose estimation. This approach combines a linear estimator that uses a minimum of 3 point correspondences with the RANSAC algorithm and an M-Estimator. The linear estimator can also be replaced by our Planar Two-point algorithm.

In Section 3.6, we approach the planar relative pose problem differently. Because the problem has only a few degrees of freedom, it becomes interesting to try a method based on the Hough Transform and discretize and analyze the whole solution space. We show that the votes for each relative pose in the discretized space can be learned from existing image data using the single point correspondence relation described in Section 3.4. In this way, the different noise characteristics are captured without explicitly modeling them. In addition, we present an efficient implementation using a precomputed lookup table, reducing the estimation process to simple lookups.

In Section 3.7, the different approaches are evaluated by applying them to both simulated data and real datasets. We compare the novel Hough Transform-based approach with the state of the art RANSAC and M-Estimator method. The state of the art method is combined with both the novel Planar Two-point algorithm, as well as the known linear estimator. In addition, an approach for general unconstrained pose estimation is applied to the data for comparison. For evaluation on real camera images, the datasets described in Appendix A are used.

In Section 3.8, we discuss the qualitative benefits of the proposed methods, as well as possibilities to improve them.

Finally, in Section 3.9, conclusions are drawn from the experiments and directions for improvement of the proposed methods are discussed.

3.2 Two-view pose estimation - related work

In this section we describe the estimation process concerning cameras that undergo planar motion. Most existing methods that deal with the planar relative pose problem are founded on the better known general unconstrained relative pose problem. We, therefore, first give a description of the unconstrained problem which is then used as a basis for the planar constrained problem.

3.2.1 The unconstrained relative pose problem

The number of degrees of freedom of the relative pose problem is 5, namely 3 for a 3D translation, plus 3 for a 3D rotation, minus 1 for the scale ambiguity. Every point correspondence in the images can remove one degree of freedom by providing one variable of information, namely 2 times 2 for the 2D pixel locations in both images, minus 3 for the unknown 3D position of the corresponding landmark. So, in order to resolve the 5 degrees of freedom, at least 5 point correspondence are needed. We must note that this is not the case if these correspondences lie in a degenerate formation, for example, if the correspondences were projections of landmarks lying on the same line in space. In such cases, one or more degrees of freedom cannot be resolved. See for example Maybank (1992); Hartley and Zisserman (2003); Kanatani (1996). We will ignore these borderline cases in this chapter.

Although 5 correspondences can resolve the degrees of freedom, they do not define a single solution, but can be the result of up to 10 distinct relative poses (Faugeras and Maybank, 1990; Maybank, 1992). A well-known algorithm that computes these solutions in closed form is the Five Point algorithm proposed by Nistér (2004); Stewénius et al. (2006). Algorithms using 5 correspondences are only useful when combined with hypothesize and test schemes, such as RANSAC. This will be explained in Section 3.5.2. Examples of robot mapping and navigation systems that use this scheme are Newman et al. (2006); Snavely et al. (2008); Fraundorfer et al. (2007); Segvic et al. (2009).

In order to determine the single solution to the unconstrained pose problem, at least 8 point correspondences are needed (Longuet-Higgins, 1981). A well-known algorithm that computes this solution is called the Eight-point algorithm (Hartley and Zisserman, 2003), which models the relative pose with a 3 by 3 matrix, known as the *essential matrix*, and applies a relatively simple least squares algorithm. Examples of robot mapping and navigation systems using this algorithm are Davison (1999); Basri et al. (1999); Zivkovic et al. (2005); Mariottini and Prattichizzo (2008); Salvi et al. (2008); Elinas and Little (2005).

One might argue that actually the Eight-point algorithm results in 4 solutions, and likewise the Five Point algorithm results in up to 40 solutions (Nistér, 2004). This is because these algorithms, like most other vision-based pose estimation methods, treat light rays falling on the camera surface as lines propagating through the back of the camera. This results in a 4 fold ambiguity (Horn, 1990). It can be easily resolved by reconstructing the landmarks for each pose and checking for which of the poses the landmarks lie in front of both cameras (Kanatani, 1996; Hartley and Zisserman, 2003).

In the field of Photogrammetry and later in Computer Vision, it is common to compare different methods for relative pose estimation. A large number of overview papers have appeared as well as literature describing comparisons with others algorithms, for example based on 6 or 7 correspondences (Stewénius et al., 2006; Brückner et al., 2008; Armangué and Salvi, 2003). They are applied to typical Computer Vision type applications, which are somewhat different than the typical Robot Vision applications. In Computer Vision, images are usually acquired by a person rotating and translating the camera in all 3 dimensions. In robotics, however, the motion is commonly constrained. Furthermore, in Computer Vision applications the camera is usually pointed towards visually interesting scenes resulting in relatively many correspondences. In robotics, the camera could be pointed to a blank wall, resulting in fewer correspondences and relatively more mismatches. In this chapter, we specifically target typical robotics applications.

3.2.2 Planar relative pose problem

The constrained planar pose problem has, evidently, fewer degrees of freedom than the unconstrained relative pose problem. The 2D rotation and 2D translation minus the scale result in 2 degrees of freedom. As we have seen one correspondence can resolve one degree of freedom. So, in general, 2 correspondences are needed to resolve the planar pose problem. However, in practice it is more common to use 3 correspondences.

Brooks et al. (1998) show that by modifying the Eight-point algorithm, a planar constrained essential matrix can be estimated using 3 point correspondences, which results in a single solution. This algorithm is known as the *Planar Three-point algorithm* and is used in various robot localization and navigation systems (Ortín and Montiel, 2001; Kosecká et al., 2005; Valgren and Lilienthal, 2008; Ramisa et al., 2009). In most of these studies, it is by the way wrongly assumed that at least 4 correspondences are needed. We will see in Section 3.4.3, where the algorithm is briefly described, that 3 correspondences suffice.

To the best of our knowledge, only two studies, namely Ortín and Montiel (2001) and Goedemé et al. (2005b), resulted in algorithms that solve the planar pose estimation problem with *two* correspondences. The first algoritm, briefly described in Ortín and Montiel (2001), is loosely based on the Three-point algorithm and uses an iterative scheme to find a solution. A proof of convergence is not given. The second one, described in Goedemé et al. (2005b), briefly describes a closed form solution. However, some of the insights necessary for implementing and using the method are omitted. The *Two-point algorithm* described in detail in Section 3.4.2 is partly based on this last closed form method.

These two studies did not investigate how many solutions 2 correspondences can define and assumed that there is always a single correct solution. As we show below, this is not the case and the proposed algorithms thus sometimes return wrong relative poses. This problem was not revealed in the performed experiments. A reason for this could be that the algorithms were combined with the RANSAC method, in order cope with noise and mismatches. The RANSAC method probably also discarded the errorneous relative pose estimates.

As opposed to the unconstrained relative pose problem, there are no papers which focus on evaluating or comparing different planar pose methods. Further, to the best of our knowledge, no study has used one of the 2 correspondence algorithm apart from the papers introducing them. In this chapter we will compare the Two-point algorithm with the Three-point algorithm and the Eight-point algorithm.

3.3 Planar pose problem and the number of solutions

We now show that two distinct planar relative poses could result in exactly the same two correspondences. This is done by first investigating the possible poses given one noise free correspondence, and then given two noise free correspondences. Specific algorithms, based on these insights, are given in the next section.

3.3.1 Solutions given 1 correspondences

Figure 3.1 depicts two robot poses: L, aligned with the world frame on the ground plane, and R somewhere else on the ground plane. The planar relative pose up to scale between these poses can be parameterized in different ways. We choose to parameterize it using two angles ϑ and ϕ. Angle ϑ denotes the direction of the translation, or heading, of robot R in the frame of robot L and angle ϕ denotes the heading of robot L in the frame of

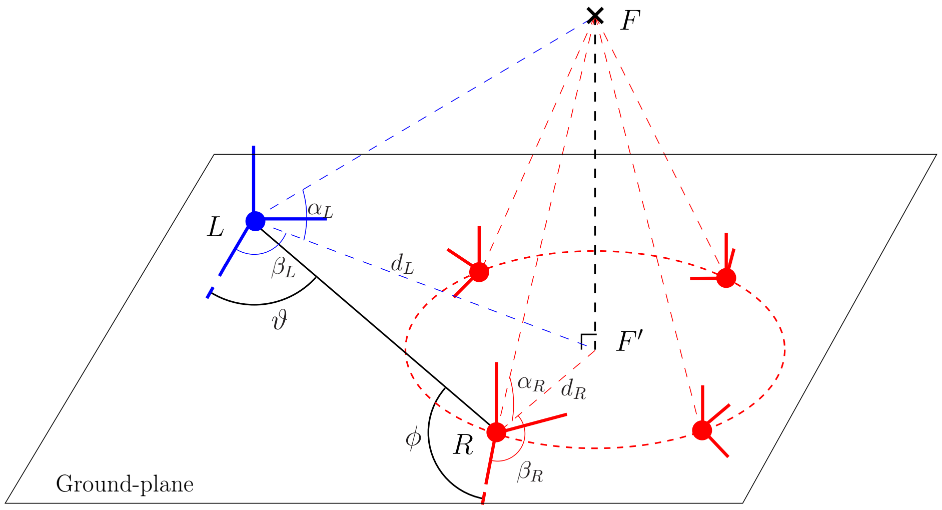
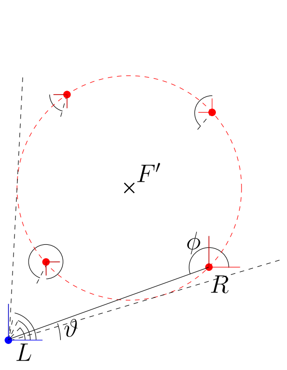
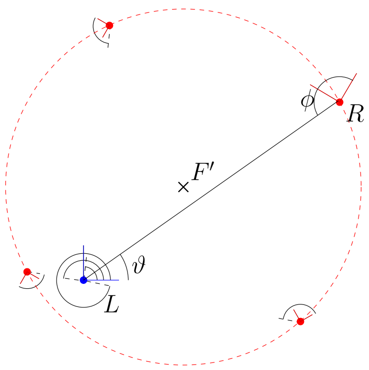


Figure 3.1: 3D visualization of two robots, L and R, positioned on a ground plane both observing a landmark F. The dashed circle on the ground plane indicates the possible positions for robot R given the pose of L and the position of F viewing angles. Four possible robot orientation for R are drawn on the circle. The angles α_L, β_L and α_R, β_R which with F is seen by the robots and the projection F' of F on the ground plane is used to compute the possible relative robot poses, modeled with angle ϑ and ϕ.



(a) $\overline{LF} > \overline{RF}$



(b) $\overline{LF} < \overline{RF}$

Figure 3.2: A top view of the 2D ground plane of robot R and L seeing landmark F. In (a) the same situation is visualized as shown in Figure 3.1, where the projection of landmark F on the ground plane F' is closer to R than L. In (b) F' is closer to L. In both situations the angles ϑ and ϕ are drawn for four possible relative robot poses.

robot R. Another common parameterization uses the heading and the rotation of robot R in the frame of robot L, such as in (Ortín and Montiel, 2001). However, the proposed parameterization reflects the symmetry of the problem.

Let us look at the way a single point correspondence constrains the possible solutions of the planar relative pose problem. Both robots, L and R observe a landmark F under a certain vertical angle α_L and α_R, and a certain horizontal angle β_L and β_R. The sinus ratio of the vertical angles α_L and α_R defines the ratio of distances from the position of L to F and the position of R to F. Because of the scale ambiguity we can fix the position of F as long as the observation angles α_L and β_L hold.

Given the landmark position F, robot pose L and observation angles α_R and β_R, we can determine the possible poses for robot R. Given the vertical observation angle α_R and the fact that the robot is confined to the ground plane, the position of the robot R has a fixed distance to F and should thus lie on a circle on the ground plane (as shown in Figure 3.1). For each position of robot R on the circle its orientation should be such that β_R points towards landmark F. In Figure 3.1, four possible positions with their appropriate orientation are drawn on the circle.

The same situation as visualized in 3D in Figure 3.1 is visualized in 2D from a top view in Figure 3.2(a). In this figure, the angles ϑ and ϕ are drawn for four possible relative robot poses. Notice that robot R is closer to landmark F than robot L. In this case, the angle ϑ is confined to a certain range of angles $\vartheta_{\text{min}} < \vartheta < \vartheta_{\text{max}}$, as indicated in Figure 3.2(a) by the dashed lines tangent to the circle. Angle ϕ, on the other hand, can take any value in the range of 0 to 2π.

Figure 3.2(b) depicts a similar situation as in Figure 3.2(a), but here robot L is closer to landmark F than robot R. In this case the circle of possible robot positions R encloses robot L. As a result the angle ϑ can now take any value in the range of 0 to 2π, while angle ϕ is confined to a certain range of angles.

3.3.2 Solutions given 2 correspondences

It is clear that one correspondence could have been the result of an infinite number of possible relative poses. As we have seen in Section 3.2.2, two correspondences will resolve all degrees of freedom, but may still result in a finite number of solutions.

Figure 3.3 visualizes the possible relative robot poses given correspondences in another way, namely by plotting ϕ as a function of ϑ. The specific function is not important right now and is derived later in Section 3.4.1. What is important is the way curves related to different landmarks intersect in the 2D space of possible relative robot poses. Say the robots observe two landmarks, F_1 and F_2. We know from Figure 3.2(a) that if robot R is closer to landmark F_1 than robot L, then ϑ is confined to a certain range of angles. This can be seen by the dashed curves in Figure 3.3(a). If the other landmark F_2 is closer to robot L, then ϕ is confined to a certain range of angles, which results in the solid curve in Figure 3.3(a).

Now, we come to an important result. Because both ϕ and ϑ are periodic, the two curves in Figure 3.3(a) intersect in at least one point. In Section 3.4.2, we will see that this is exactly one intersection, giving a single solution to the robot pose problem. If, however, both landmarks are closer to robot R, then both curves are confined to a certain range of angles ϑ and must intersect in at least two points (see Figure 3.3(b)). The same holds for the situation where the two landmarks are closer to L. We will see in Section 3.4.2 that there are exactly two intersections.

Given the spatial arrangement of the robot and landmark positions, two point correspondences resulting from these landmarks define 1 or 2 possible relative robot poses:

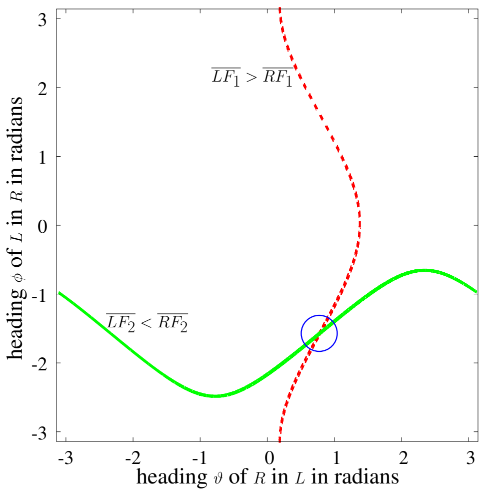
$$\left(\overline{LF_1} - \overline{RF_1}\right)\left(\overline{LF_2} - \overline{RF_2}\right) = \begin{cases} > 0 & \rightarrow \quad \text{1 solution} \\ < 0 & \rightarrow \quad \text{2 solutions,} \end{cases} \tag{3.1}$$

for which F_1 and F_2 denote the positions of the two landmarks and $\overline{XY}$ denotes the distance from the position of pose X to the landmark Y.

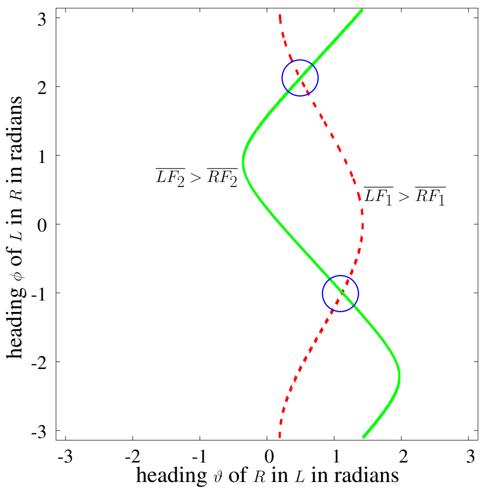
Thus, in practice if an algorithm is given 2 noise free correspondences, yet is designed to always return a single relative pose, then this pose will be wrong in 25% of the cases. What is needed is an algorithm that computes both solutions if there are two, which then can be post-processed by an hypothesize and test algorithm.

3.4 Algorithms for planar pose estimation

We begin by deriving the function that relates the heading of the first robot with respect to the second ϑ with the heading for the second robot with respect to the first ϕ, given a single point correspondence. Using this function we derive an algorithm which uses 2 point correspondence to compute the solutions to the problem, sometimes returning a single solution and sometimes two solutions. In this section we also give a short description of a known algorithm that uses 3 point correspondences and approaches the problem by formulating a set of linear equations which are solved using a least squares technique.



(a)



(b)

Figure 3.3: Visualization of the possible relative robot poses by plotting the heading ϕ as a function of ϑ for different point correspondences. In (a) one landmark is closer to R and another is closer to L, resulting in two curves intersecting in a single point. In (b) both landmarks are closer to R, resulting in two intersections.

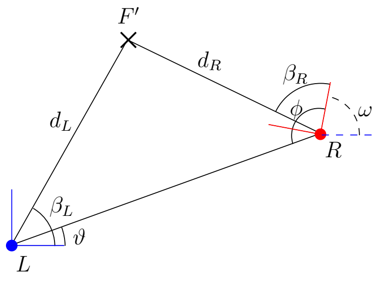


Figure 3.4: A top view of the 2D ground plane of robot R and L and the projection of a landmark on this plane F'. The relative pose between R and L is parameterized using ϑ and ϕ. The robots see the landmark with the horizontal observation angles β_L and β_R. The ratio of the distance d_L and d_R can be derived from the vertical observation angles. The relative rotation ω is used in deriving the 3-point algorithm in Section 3.4.3.

It is again assumed that image point correspondences are obtained by a noise free projection of landmarks (without mismatches). The point correspondences are denoted by two sets of 3D vectors of unit length $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and $\{\mathbf{x}'_1, \ldots, \mathbf{x}'_n\}$, with $\mathbf{x}_i = [x_i, y_i, z_i]^T$, corresponding to n landmarks seen in respectively robot L and R. This corresponds to a projection on a sphere of radius 1 around the viewpoint of the camera, which allows the application of the algorithms on different types of omnidirectional vision systems as well as conventional cameras (Bunschoten, 2003; Mariottini and Prattichizzo, 2008).

3.4.1 One-point mapping function

A function that relates ϕ and ϑ to a point correspondence can be based solely on trigonometry. First, the 3D problem as shown in Figure 3.1 is reduced to a simpler 2D problem. In Section 3.3, we already pointed out that the vertical observation angles α_L and α_R can be used to determine the ratio of distances d_L of L to projection of the landmark on the ground plane F' and d_R of R to F'. The length of F to F' can be expressed using α_L and d_L and using α_R and d_R (see Figure 3.1):

$$\overline{FF'} = d_L \tan(\alpha_L) = d_R \tan(\alpha_R), \tag{3.2}$$

which results in the following ratio of d_L and d_R:

$$\frac{d_L}{d_R} = \frac{\tan(\alpha_R)}{\tan(\alpha_L)}. \tag{3.3}$$

The angles α_* can be computed from the z coordinates of the spherical image points using

$$\alpha_* = \arcsin(z) \tag{3.4}$$

Figure 3.4 shows the triangle defined by L, R and F' on the ground plane. The angle $\angle RLF'$ is simply defined by the horizontal observation angle β_L minus ϑ. The angles β_* can be computed from the x and y coordinates of the image points using

$$\beta_* = \text{atan2}(y, x), \tag{3.5}$$

where the function atan2(y, x) is a variation of the arctangent function, as implemented in standard C, which returns an angle in the range $\langle -\pi, \pi]$.

Angle $\angle F'RL$ can then be found using the Law of Sines.

$$\begin{aligned} \frac{\sin(\angle F'RL)}{d_L} &= \frac{\sin(\beta_L - \vartheta)}{d_R}, \\ \angle F'RL &= \begin{cases} \arcsin\left(\frac{d_L}{d_R}\sin(\beta_L - \vartheta)\right) \\ \pi - \arcsin\left(\frac{d_L}{d_R}\sin(\beta_L - \vartheta)\right) & \text{if } d_L > d_R, \end{cases} \end{aligned} \tag{3.6}$$

where we explicitly take the ambiguous case into account that d_R is smaller than d_L.

Angle ϕ can be determined by adding the horizontal observation angle β_R, and using Equation (3.3):

$$\phi = \begin{cases} \beta_R + \arcsin\left(\frac{\tan(\alpha_R)}{\tan(\alpha_L)}\sin(\beta_L - \vartheta)\right) \\ \beta_R + \pi - \arcsin\left(\frac{\tan(\alpha_R)}{\tan(\alpha_L)}\sin(\beta_L - \vartheta)\right) & \text{if } \tan(\alpha_R) > \tan(\alpha_L). \end{cases} \tag{3.7}$$

We now find a function that expresses the angle ϕ as a function of angle ϑ. As we saw in Section 3.3.2 one value for ϑ sometimes defines two possible values for ϕ (see for example the dashed line in Figure 3.3(a)). We can also express angle ϑ as a function of angle ϕ. The derivation of this function is analogous to that of Equation (3.7) and clearly shows the symmetry of the chosen parameterization:

$$\vartheta = \begin{cases} \beta_L + \arcsin\left(\frac{\tan(\alpha_L)}{\tan(\alpha_R)}\sin(\beta_R - \phi)\right) \\ \beta_L + \pi - \arcsin\left(\frac{\tan(\alpha_L)}{\tan(\alpha_R)}\sin(\beta_R - \phi)\right) & \text{if } \tan(\alpha_L) > \tan(\alpha_R). \end{cases} \tag{3.8}$$

We have now obtained functions that map ϑ to ϕ given a single point correspondence. Using this function, we can draw curves of possible robot poses as shown previously in Figure 3.3. In Section 3.6, we will see how this result can be used to compute a robust estimate given multiple point correspondences.

3.4.2 Planar Two-point algorithm

This section gives the complete derivation of the novel Two-point algorithm. The derivation is again based solely on trigonometry.

We assume the two robots L and R both observe two landmarks, F_1 and F_2 (see Figure 3.5). Robot L observes the landmarks with angles $(\alpha_{L1}, \beta_{L1})$ and $(\alpha_{L2}, \beta_{L2})$ and robot R with angles $(\alpha_{R1}, \beta_{R1})$ and $(\alpha_{R2}, \beta_{R2})$. The goal is to determine 1 or 2 relative poses parameterized as (ϑ, ϕ) pairs. Figure 3.5 shows all variables involved in the derivation of the algorithm.

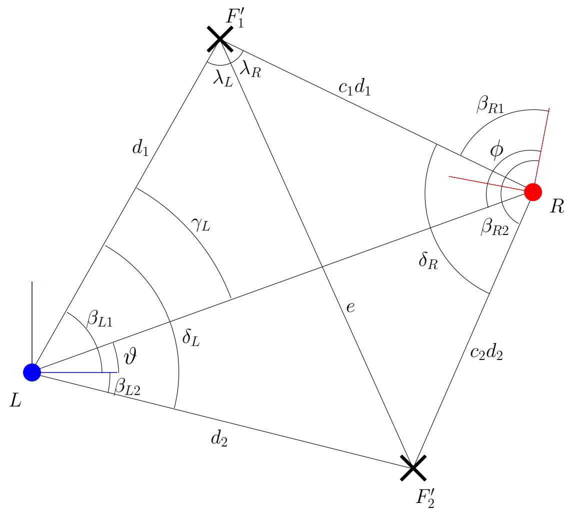


Figure 3.5: Schematic drawings showing the variables involved in the two-point algorithm. The scene is again seen from a top-view in which distances and angles are measured only in 2D (parallel to the ground-floor).

An intuitive approach to this problem is to define two functions using Equation (3.7) and try to substitute one in the other:

$$\beta_{R1} + \arcsin\left(\frac{\tan(\alpha_{R1})}{\tan(\alpha_{L1})}\sin(\beta_{L1} - \vartheta)\right) = \beta_{R2} + \arcsin\left(\frac{\tan(\alpha_{R2})}{\tan(\alpha_{L2})}\sin(\beta_{L2} - \vartheta)\right). \tag{3.9}$$

It is difficult, if not impossible, to solve ϑ using this function. Therefore, we take a different approach. We will try to find enough parameters of the quadrangle (also known as a tetragon) formed by the points L-F_2'-R-F_1' (as seen in figure 3.5), to compute the angles of a triangle between one of the landmarks and the robot poses. If for example we know the angle $\angle RLF_1'$, which is denoted by γ in Figure 3.5, it is straightforward to determine ϑ.

We can compute the angles of the quadrangle at L and R, denoted by δ_L and δ_R, by taking the differences between both pairs of horizontal observation angles β's:

$$\delta_L = \beta_{L2} - \beta_{L1} \tag{3.10}$$
$$\delta_R = \beta_{R2} - \beta_{R1} \tag{3.11}$$

The distances from the two views to the landmarks cannot be determined, but the ratio between these distances can be determined using the vertical view angles as described previously (see Equation (3.3)). Here, we introduce extra variables c_1 and c_2 for convenience:

$$c_1 = \frac{c_1 d_1}{d_1} = \frac{\tan(\alpha_{L1})}{\tan(\alpha_{R1})} \tag{3.12}$$

$$c_2 = \frac{c_2 d_2}{d_2} = \frac{\tan(\alpha_{L2})}{\tan(\alpha_{R2})}. \tag{3.13}$$

We now have enough parameters to determine the shape of the quadrangle, except for the scale which cannot be determined from point correspondences. Without loss of generality, we set one of the distances, namely the one from view L to landmark F_1', to 1:

$$d_1 \equiv 1 \tag{3.14}$$

The ratio between distances from view L to landmarks F_1' and F_2' can be computed with the use of the distance between landmark F_1' and F_2' denoted by e. The distance e can be expressed in two ways using the left triangle L-F_1'-F_2' and the right triangle R-F_1'-F_2' by applying the Law of Cosines:

$$e^2 = 1 + d_2^2 - 2d_2\cos(\delta_L) \tag{3.15}$$

$$e^2 = c_1^2 + c_2^2 d_2^2 - 2c_1c_2d_2\cos(\delta_R) \tag{3.16}$$

Equating the right parts of equations 3.15 and 3.16, gives a second order polynomial in d_2:

$$1 + d_2^2 - 2d_2\cos(\delta_L) = c_1^2 + c_2^2 d_2^2 - 2c_1c_2d_2\cos(\delta_R) \tag{3.17}$$

$$\left(1 - c_2^2\right) d_2^2 - 2\left(\cos(\delta_L) - c_1c_2\cos\left(\delta_R\right)\right) d_2 + 1 - c_1^2 = 0 \tag{3.18}$$

This can be solved using the quadratic formula:

$$a = 1 - c_2^2 \tag{3.19}$$

$$b = -2(\cos(\delta_L) - c_1c_2\cos(\delta_R)) \tag{3.20}$$

$$c = 1 - c_1^2 \tag{3.21}$$

$$d_2 = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{3.22}$$

This could result in two real solutions for d_2, which indicates that given two point correspondences there could be two possible solutions for the relative pose. The rest of the derivation has to be performed using both these values.

Using the distances d_1 and d_2 we can calculate distance between the two landmarks:

$$e = \sqrt{1 + d_2^2 - 2d_2\cos(\delta_L)} \tag{3.23}$$

Using once more the Law of Cosines the angles λ_L and λ_R can be calculated:

$$\lambda_L = \begin{cases} \arccos\left(\frac{d_2^2 - e^2 - 1}{-2e}\right) & \text{if } \delta_L < \pi \\ -\arccos\left(\frac{d_2^2 - e^2 - 1}{-2e}\right) & \text{if } \delta_L > \pi \end{cases} \tag{3.24}$$

$$\lambda_R = \begin{cases} \arccos\left(\frac{c_2^2 d_2^2 - e^2 - c_1^2}{-2c_1 e}\right) & \text{if } \delta_R > \pi \\ -\arccos\left(\frac{c_2^2 d_2^2 - e^2 - c_1^2}{-2c_1 e}\right) & \text{if } \delta_R < \pi, \end{cases} \tag{3.25}$$

where we took into account that the triangles can be acute or obtuse. The angle L-F'_1-R, which we denote with λ, is given by

$$\lambda = \lambda_L + \lambda_R. \tag{3.26}$$

From the angle λ we can compute γ_L using the atan2 function to get an angle in the range $\langle -\pi, \pi]$:

$$\gamma_L = \text{atan2}(c_1 \sin(\lambda), 1 - c_1 \cos(\lambda)) \tag{3.27}$$

Using angle β_{L1} we can now compute ϑ:

$$\vartheta = \beta_{L1} - \gamma_L \tag{3.28}$$

The angle ϕ is computed using the triangle $L - F'_1 - R$ and angle β_{R1}:

$$\phi = \beta_{R1} + (\pi - \gamma_L - \lambda) \tag{3.29}$$

The complete algorithm as given by equations (3.10)-(3.13) and (3.19)-(3.29) can be applied to every pair of distinct point correspondences. However, for a robust implementation which handles noisy point correspondences and mismatches one has to be cautious. Most implementations of the arccosine function return a complex number if it is called with a variable which has an absolute value higher than 1. This could happen for some noisy pair of point correspondences. Furthermore, it could happen that c_1 or c_2 becomes negative if a point correspondence has a positive vertical angle in one view and a negative vertical angle in the other. In both of these cases, the relative pose is undetermined and the algorithm will output no solutions.

3.4.3 Planar Three-point algorithm

We now give a short description of a known algorithm that uses 3 point correspondences (Ortín and Montiel, 2001; Brooks et al., 1998). This algorithm is based on the well-known Eight-point algorithm, which can be used to estimate unconstrained camera motion (Longuet-Higgins, 1981; Hartley and Zisserman, 2003).

The camera motion is parameterized using a 3x3 matrix E called the essential matrix. It can be constructed from a 3x3 rotation matrix R and a 3D translation vector $\mathbf{t} = [t_x, t_y, t_z]^T$:

$$E = [\mathbf{t}]_\times R, \tag{3.30}$$

in which $[\mathbf{t}]_\times$ is the matrix representation of the cross product with $\mathbf{t}$,

$$[\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}. \tag{3.31}$$

An important characteristic of the essential matrix is that it relates point correspondences using the following linear relation:

$$(\mathbf{x}'_i)^T E \mathbf{x}_i = 0 \qquad \text{for all } i. \tag{3.32}$$

This can be used to estimate the nine entries of E from a set of 8 or more point correspondences using the Direct Linear Transform (DLT), which works as follows (Hartley and Zisserman, 2003). First, the linear equations are rewritten in the following form:

$$A\mathbf{e} = 0 \tag{3.33}$$

$$\begin{bmatrix} x_1x'_1 & \cdots & x_nx'_n \\ x_1y'_1 & \cdots & x_ny'_n \\ x_1z'_1 & \cdots & x_nz'_n \\ y_1x'_1 & \cdots & y_nx'_n \\ y_1y'_1 & \cdots & y_ny'_n \\ y_1z'_1 & \cdots & y_nz'_n \\ z_1x'_1 & \cdots & z_nx'_n \\ z_1y'_1 & \cdots & z_ny'_n \\ z_1z'_1 & \cdots & z_nz'_n \end{bmatrix}^T \begin{bmatrix} e_{11} \\ e_{12} \\ e_{13} \\ e_{21} \\ e_{22} \\ e_{23} \\ e_{31} \\ e_{32} \\ e_{33} \end{bmatrix} = 0, \tag{3.34}$$

where e_{rc} is the entry of matrix E in row r and column c.

The number of degrees of freedom of E, and thus also $\mathbf{e}$, is 8, namely 9 for the 9 matrix entries minus 1 for the scale ambiguity. Thus, at least 8 linear equations are needed and, because each point correspondence provides one equation, at least 8 point correspondences are needed. A least square solution for E:

$$\arg\min_E \sum_i (\mathbf{x}'_i)^T E \mathbf{x}_i = 0 \tag{3.35}$$

can be found by applying a singular value decomposition (SVD) to $A^T A$ and taking the eigenvector corresponding to the smallest singular value.

The essential matrix has rank 2 and its first two eigenvalues are equal (Hartley and Zisserman, 2003). The DLT algorithm does not enforce this nonlinear constraint. By applying an SVD to E giving $E = USV'$ and building a new matrix $E' = U\text{diag}(1, 1, 0)V'$ we find an essential matrix that does enforce this constraint and is normalized at the same time. For details see Hartley and Zisserman (2003).

In case of a planar camera motion, 5 of the 9 entries of the essential matrix are zero by definition. This can be seen by expressing E in the heading ϑ and the rotation angle ω:

$$\begin{aligned} E &= \begin{bmatrix} \cos(\vartheta) \\ \sin(\vartheta) \\ 0 \end{bmatrix}_\times \begin{bmatrix} \cos(\omega) & -\sin(\omega) & 0 \\ \sin(\omega) & \cos(\omega) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & \sin(\vartheta) \\ 0 & 0 & -\cos(\vartheta) \\ \sin(\omega - \vartheta) & \cos(\omega - \vartheta) & 0 \end{bmatrix} \end{aligned} \tag{3.36}$$

This formula is a common way to specify the planar constrained essential matrix (Ortín and Montiel, 2001; Brooks et al., 1998; Bunschoten, 2003). The rotation angle ω relates to ϕ and ϑ by

$$\omega = \pi + \vartheta - \phi, \tag{3.37}$$

as can be seen in Figure 3.4. Using this we can rewrite Equation (3.36) into

$$E = \begin{bmatrix} 0 & 0 & \sin(\vartheta) \\ 0 & 0 & -\cos(\vartheta) \\ \sin(\phi) & -\cos(\phi) & 0 \end{bmatrix}, \tag{3.38}$$

which clearly shows the symmetric relation between angles ϑ and ϕ.

Only the non-zero entries of E have to be estimated reducing Equation (3.34) to:

$$\begin{bmatrix} x_1 z'_1 & \cdots & x_n z'_n \\ y_1 z'_1 & \cdots & y_n z'_n \\ z_1 x'_1 & \cdots & z_n x'_n \\ z_1 y'_1 & \cdots & z_n y'_n \end{bmatrix}^T \begin{bmatrix} e_{13} \\ e_{23} \\ e_{31} \\ e_{32} \end{bmatrix} = 0, \tag{3.39}$$

Computing a least square solution is similar to the non planar case, except for the minimal number of correspondences. The number of degrees of freedom of this reduced E is 3, namely 4 for the 4 non-zero entries minus 1 for the scale ambiguity. Thus, only 3 point correspondences are necessary. In the next sections, when dealing with noisy correspondences, we examine the error function that this least squares solution minimizes.

Now that we found the essential matrix E, we can extract the rotation R and the translation $\mathbf{t}$. Each essential matrix can be decomposed in 4 possible relative robot poses (Horn, 1990; Hartley and Zisserman, 2003). This ambiguity can be resolved by reprojecting the point correspondences to the world domain and checking if their reprojection has a positive depth in both cameras, i.e. if they lie in front of the camera surfaces where the image points were found. For the correct relative pose this will indeed be the case.

To summarize, by estimating the essential matrix a method is obtained for estimating the planar relative pose problem using 3 point correspondences. Actually, the 3 correspondences overdetermine the estimation problem which has only 2 degrees of problem as we have shown. Thus, a least square technique is used to find a solution that best fits the 3 correspondences.

3.5 Estimation in the presence of noise and mismatches: state of the art

In the previous sections we assumed that image point correspondences were obtained by a noise free projection of 3D landmarks. As we have explained in Section 3.3, from three or more of these correspondences the camera pose can be determined. In Figure 3.6(a), the curves of a few noise free correspondences are plotted in the camera pose solution space and, as can be seen, all of them intersect in the same pose. In practice, image point correspondences are automatically found by an image matching algorithm such as the SIFT method (Lowe, 2004) as explained in Chapter 2. Each image point position is therefore subject to a considerable amount of noise. On top of that, part of the point correspondences are the result of mismatched image points. Figure 3.6(b) shows that the curves in pose space resulting from these noisy correspondences do not intersect in a single point. The question is how to determine the camera pose given these noisy correspondences.

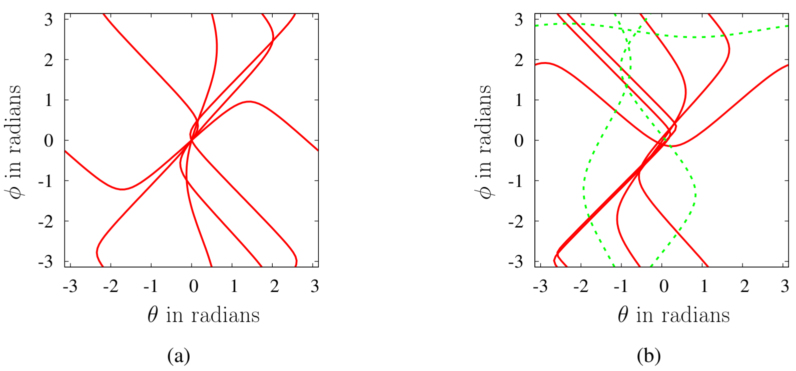


Figure 3.6: Visualization of the possible relative robot poses by plotting ϕ as a function of ϑ for multiple correspondences randomly picked using a relative pose with $\vartheta = \phi = 0$. (a) Generated using 5 noise free correspondences. (b) Generated using 5 noisy correct correspondences (solid curves) and 3 mismatches (dashed curves). Note that, by coincidence, one of the mismatches corresponds to a curve close to the actual pose.

The problem can be formulated as searching for the *Maximum Likelihood*. Given n point correspondences denoted as $\{\xi_1, \ldots, \xi_n\}$. Each correspondence is parametrized by the angles $\xi_i = (\alpha_{Li}, \beta_{Li}, \alpha_{Ri}, \beta_{Ri})$. The problem is then to determine the most likely relative pose $\vartheta_{\text{ML}}, \phi_{\text{ML}}$:

$$(\vartheta_{\text{ML}}, \phi_{\text{ML}}) = \arg\max_{(\vartheta,\phi)} p(\xi_1, \ldots, \xi_n | \vartheta, \phi) \tag{3.40}$$

$$= \arg\min_{(\vartheta,\phi)} \sum_i -\log p(\xi_i | \vartheta, \phi), \tag{3.41}$$

where we make the common assumption that the observed point correspondences are independent given the robot pose.

The term $-\log p(\xi_i|\vartheta, \phi)$ describes the negative log likelihood, and is known as the *error function*. The major problem when estimating the camera pose, is that this error function is not known. Various assumptions about the error function have resulted in different approaches and algorithms. A common approach is to try to describe the noise of the correspondences that resulted from correct matches with an error function and then treat the correspondences that have a very large error as the mismatches.

In this section we explain a state of the art (Torr and Murray, 1997; Hartley and Zisserman, 2003) method to deal with this problem in the context of planar relative pose estimation. The method uses explicit error functions for the correspondences resulting from correctly matched image points, given in Subsection 3.5.1. Mismatches are dealt with using the RANSAC algorithm to provide an initial estimate of the pose, as explained in Subsection 3.5.2. This is followed by an M-Estimator to improve this estimate, as explained in Subsection 3.5.3. The method can be combined with both the Planar Two-point algorithm described in Section 3.4.2 and the Planar Three-point algorithm described in Section 3.4.3.

3.5.1 Error functions for correct matches

We now describe standard error functions that return the error of a measured point correspondence and an estimated planar pose, given that the correspondence resulted from a correct match. They focus on the noise that was caused by the projection process of the 3D landmarks on the image surface. This is usually assumed to be isotropic homogeneous Gaussian distributed noise. The negative log likelihood is then defined as the sum of the Euclidean distance between the image points of the correspondence and the noise free projections of the actual 3D landmark. This measure is known as the *geometric error* (Hartley and Zisserman, 2003). Unfortunately, we cannot compute it, because we do not know the actual position of this landmark.

What we can easily compute is the so-called *algebraic error* (Hartley and Zisserman, 2003), which was actually already defined when writing the pose estimation problem as a linear estimation problem in Section 3.4.3. Specifically, it is the residual of the least square error function defined in Equation (3.39):

$$r_i = x_i z_i' e_{13} - y_i z_i' e_{23} + z_i x_i' e_{31} - z_i y_i' e_{32} \tag{3.42}$$

Although this measure is easy to compute, it does not have a geometrical meaning. That is, it is not related to the projection process of the imaging system. However, it can be used to compute a first order approximation of the geometric error, which is known as the *Sampson distance* (Torr and Murray, 1997). This is done by weighing it with the size of its gradient with respect to the image point coordinates. For the used spherical image representation, we assume Gaussian noise for all three axis. The gradient is then given by:

$$\nabla r_i = \begin{pmatrix} \frac{\partial r_i}{\partial \mathbf{x}} \\ \frac{\partial r_i}{\partial \mathbf{x}'} \end{pmatrix} = \begin{pmatrix} z_i' e_{13} \\ z_i' e_{23} \\ x_i' e_{23} - y_i' e_{32} \\ z_i e_{31} \\ z_i e_{32} \\ x_i e_{32} - y_i e_{23} \end{pmatrix}, \tag{3.43}$$

which are determined using Equation (3.42).

The Sampson distance is now given by:

$$d_i = \frac{r_i}{|\nabla r_i|}, \tag{3.44}$$

where $|\nabla r_i|$ denotes the Euclidean norm of ∇r_i.

3.5.2 RANSAC

A method that is particularly robust against a high percentage of mismatches is the *RANSAC* algorithm (RANdom SAmple Consensus) (Fischler and Bolles, 1981; Torr and Murray, 1997), which is nowadays the standard method for relative pose estimation from

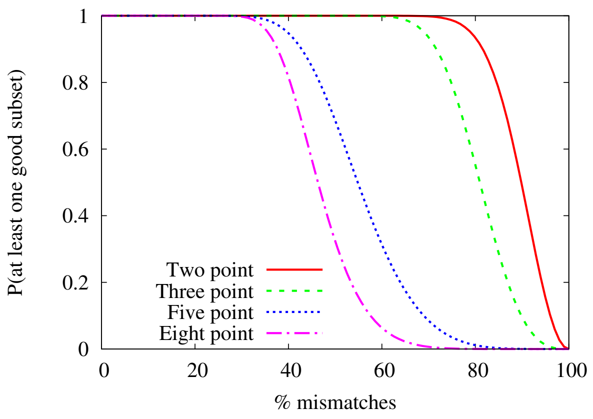


Figure 3.7: Comparison of the performance of the RANSAC when using eight, five, three or two point correspondences, computed using Equation (3.45) while varying the percentage of mismatches. The number of candidates N was set to the typical value of 100.

images. RANSAC randomly picks a subset, or sample, of correspondences from which it computes a candidate relative pose using one of the closed form estimators. An exmample is the Two-point algorithm (Section 3.4.2). For each correspondence of the complete set we check if it fits the candidate pose, by computing its error - for example using the Sampson distance - and checking if it is below a preset threshold. A correspondence that fits the candidate pose is said to be an *inlier*. This procedure is repeated for different random samples producing a large number of candidate poses. Finally, from these poses the one is chosen with the highest number of inliers.

The rationale of RANSAC is that at least one of the candidate poses is computed from only correct point correspondences and will thus be close to the maximum likelihood, which results in a relatively large number of inliers. The probability p of RANSAC picking at least one sample of only correct matches, can be related to the number of samples N, the percentage of correct correspondences w, the size of the sample s and the average number of solutions given by the closed form algorithm k:

$$p = 1 - (1 - w^s)^{\left(\frac{N}{k}\right)} . \tag{3.45}$$

This is a variation of the formula given by Fischler and Bolles (1981), which lacks the dependence on k. As can be seen, p depends exponentially on s. So it is beneficial to choose it as small as possible. That is why it is important to use an algorithm that needs the minimal number of correspondences. The value of k, which is usually ignored when determining p (Fischler and Bolles, 1981; Hartley and Zisserman, 2003; Torr and Murray, 1997), is 1.5 for the Two point algorithm (see Section 3.3.2), 2.74 for the Five Point algorithm following (Nistér, 2004) and 1 for the others.

In Figure 3.7, Equation (3.45) is used to show the advantage of using an algorithm that uses two correspondences instead of to the standard algorithm that uses three correspondences. As can be seen, the largest gain of using a Two-point algorithm is for a high percentage of mismatches between 70% and 95%. Note, however, that the formula does not take into account image noise. In practice correct point correspondences are noisy and therefore pose estimates based on fewer correspondences are less accurate. This is not taken into account in Equationr 3.45 or Figure 3.7.

The final RANSAC solution is based on only the minimal set of noisy correspondences. Usually the RANSAC method is followed by an iterative algorithm that uses the solution of RANSAC as the initial guess.

3.5.3 M-Estimators

All the inliers of the RANSAC solution can be used to compute a new pose using a closed form algorithm that can handle an unlimited number of point correspondences, such as the Planar Three-point algorithm. However, as we have seen, the Planar Three-point algorithm minimizes the algebraic error instead of the geometric error. We can use the size of the gradient of the residual with respect to the image point coordinates to make an improved estimate of the essential matrix E^* by computing:

$$E^* = \arg\min_E \sum_i \frac{1}{|\nabla r_i|}(\mathbf{x}_i')^T E \mathbf{x}_i = 0. \tag{3.46}$$

As can be seen, E^* depends on ∇r_i, which in turn depends on an estimate of E. Therefore, the essential matrix is estimated using an iterative scheme, which is termed iteratively reweighted least squares (IRLS).

Given new estimates of E we should again determine which of the correspondences are inliers by computing their error, for example, by using the Sampson Distance. Usually, instead of setting a hard error threshold, a sliding scale is used to weight the correspondences. A common method to weight correspondences is known as *Huber Weighting* (Torr and Murray, 1997):

$$w_i = \begin{cases} 1 & d_i < \sigma \\ \sigma/d_i & \sigma < d_i < 3\sigma \\ 0 & 3\sigma < d_i \end{cases} \tag{3.47}$$

By combining this weight with the size of the gradient, we obtain the following update formula:

$$E^* = \arg\min_E \sum_i w_i \frac{1}{|\nabla r_i|}(\mathbf{x}_i')^T E \mathbf{x}_i = 0. \tag{3.48}$$

The combination of such robust weighting schemes with IRLS (Equation (3.46)) is called an *M-Estimator*.

3.6 Estimation in the presence of noise and mismatches: novel histogram-based estimator

In this section, we describe a completely novel robust method to estimate a planar pose given noisy correspondences including mismatches. The method is based on discretizing

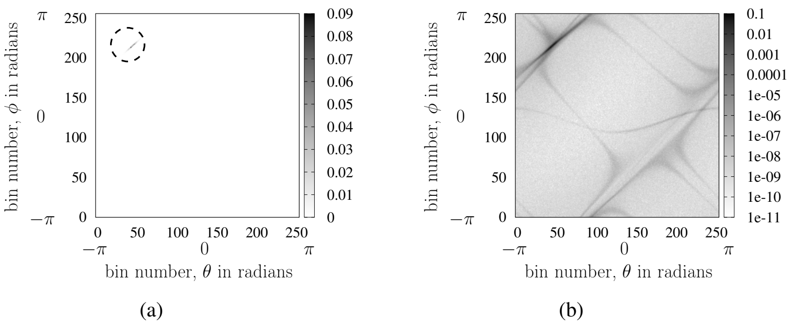


Figure 3.8: Example discretized likelihood of the relative pose space (a) and log likelihood (b), given 5 correct matches and 5 mismatches. White bin indicate low likelihood, while dark bins indicate high likelihood. Both ϑ, ϕ and c are discretized into 256 bins, each approximately 0.025 radians wide. The model used for generating this likelihood assumed 90% mismatches and 0.01 projection noise, as explained in Section 3.6.3. For clarity, the dashed circle indicates the maximum likelihood in (a).

the solution space and is, in this respect, similar to the well-known Hough Transform. For each bin in the discretized space, the likelihood of the observed correspondences is computed (Subsection 3.6.1). Instead of computing the likelihood for each correspondence on the fly, it is taken from a precomputed lookup table (LUT) (Subsection 3.6.2), which represents the likelihood in a discretized way. The LUT is kept small by taking advantage of the One-point mapping function described in Section 3.4.1. The likelihood could be approximated using an error measurement function described in Subsection 3.5.1. We, however, construct it from existing sensor measurements and groundtruth pose information or using a simulator modeling the planar pose problem (Subsection 3.6.3).

3.6.1 Discretizing the solution space

We go back to the problem description formulated as finding the Maximum Likelihood in Equation (3.40). Instead of evaluating the likelihood of certain selected poses, like in RANSAC, or determining the gradient of the likelihood to improve such a pose, like the M-Estimator does, we now take a very different approach.

Like in the well-known Hough Transform, we discretize the solution space into a grid. In our case, this is a 2D space of relative planar poses represented by the angles ϑ and ϕ. Then, the likelihood for each of the poses corresponding to the grid points is determined given the set of observations. In this way, we get a *discretized likelihood* over the complete space of poses. The pose with the highest value approximates the maximum likelihood solution. See Figure 3.8 for an example of such a discretized likelihood, given noisy correspondences including mismatches.

Because we assume the noise of the different observed point correspondences is independent given the relative pose, the likelihood of a set of correspondences can be computed by multiplying the likelihoods of each single correspondence (see Equation (3.41)). Actually, in practice we should restrain from multiplying likelihoods, because of the risk of underflowing machine precision. Rather, we work in log space, summing the log of the likelihoods. This is similar to the Hough Transform, which sums the so-called votes from each correspondence. As opposed to the Hough Transform, in our approach these votes are not 0 or 1, but we use the log likelihood to weight the votes. The question now is how to determine the log likelihood of the different poses given a single correspondence.

3.6.2 Lookup table

For each point correspondence, we need to determine a log likelihood for all the poses in the discretized pose space. Even if determining such a log likelihood would cost only a limited amount of computational time, the overall computational cost quickly grows prohibitively large. So in practice, this rules out the use of evaluating functions as described previously in Section 3.5.1.

To solve this problem we discretize, besides all possible poses, all possible observations and precompute a lookup table of the log likelihood for each combination of observation and pose. For each new observation we can then quickly lookup the log likelihood values for the different poses.

The log likelihood of a single correspondence $\xi = (\alpha_L, \beta_L, \alpha_R, \beta_R)$ is given by $-\log p(\alpha_L, \beta_L, \alpha_R, \beta_R | \vartheta, \phi)$. Thus, if we would naively construct such a lookup table, it would have 6 dimensions, 4 for the point correspondence angles and 2 for the relative pose. This would be quite impractical. In order to keep the size of the LUT comprehensible, the 6D space should be discretized in large bins, resulting in a large discretization error. We now show that the dimensionality of the LUT can be reduced to only 3 dimensions by using the One-point mapping function introduced in Section 3.4 and some common assumptions about the noise.

The One-point mapping function given in Equation (3.7) can be written as

$$\phi - \beta_R + \arcsin\left(\frac{\tan(\alpha_R)}{\tan(\alpha_L)}\sin(\vartheta - \beta_L)\right) \approx 0. \tag{3.49}$$

As can be seen, the terms α_L and α_R are only used in the combination $\frac{\tan(\alpha_R)}{\tan(\alpha_L)}$. In addition, variables ϑ and β_L and variables ϕ and β_R are only used in the combinations $\vartheta - \beta_L$ and $\phi - \beta_R$. They describe the horizontal angles to the landmark relative to the heading of the cameras. The joint probability of observing a correspondence under a certain relative pose can thus be represented as:

$$p(\alpha_L, \beta_L, \alpha_R, \beta_R, \vartheta, \phi) = p\left(\frac{\tan(\alpha_R)}{\tan(\alpha_L)}, \vartheta - \beta_L, \phi - \beta_R\right). \tag{3.50}$$

This holds if we assume that the noise characteristic of the horizontal and vertical view angles of a point correspondence does not depend on actual value of that point correspondence. This assumption is comparable with the common assumption that the noise of each image point coordinate does not depend on the image point location itself.

Because of the symmetry of the representation of the relative planar pose, the two points of the point correspondences and the heading angles can be swapped giving the same result:

$$p\left(r,\vartheta-\beta_L,\phi-\beta_R\right)=p\left(\frac{1}{r},\phi-\beta_R,\vartheta-\beta_L\right), \tag{3.51}$$

where we introduced $r=\frac{\tan(\alpha_R)}{\tan(\alpha_L)}$ for convenience. In practice, this means that we only have to construct a LUT for $0<r<1$ and swap $\vartheta-\beta_L$ with $\phi-\beta_R$ and use $\frac{1}{r}$ instead of r if $r>0$.

The likelihood can be determined from the joint probability by dividing by the probability of the pose $p(\vartheta,\phi)$:

$$\begin{aligned} p(\xi|\vartheta,\phi) &= p(\alpha_L,\beta_L,\alpha_R,\beta_R|\vartheta,\phi) && (3.52)\\ &= p(\alpha_L,\beta_L,\alpha_R,\beta_R,\vartheta,\phi)/p(\vartheta,\phi) && (3.53)\\ &= p\left(\frac{\tan(\alpha_{Ri})}{\tan(\alpha_{Li})},\vartheta-\beta_{Li},\phi-\beta_{Ri}\right)/p(\vartheta,\phi). && (3.54)\end{aligned}$$

Usually, during the construction of the LUT, one takes care that the different relative poses are uniformly distributed, making the likelihood proportional to the joint probability:

$$p(\xi|\vartheta,\phi)\propto p\left(\frac{\tan(\alpha_R)}{\tan(\alpha_L)},\vartheta-\beta_L,\phi-\beta_R\right). \tag{3.55}$$

It is now straightforward to construct a LUT from one of the error functions described in Section 3.5.1, by evaluating the function for the different poses and point correspondences. However, those measures do not take mismatches into account. In Section 3.6.3, we will show how to construct a LUT from existing data including mismatches. Figure 3.9 visualizes an example 3D lookup table that was constructed using data from a simulator.

The LUT can now be used to efficiently determine the bin of the most likely relative pose given a set of correspondences. For each correspondence ξ_i, we compute the value of $\frac{\tan(\alpha_{Ri})}{\tan(\alpha_{Li})}$ and pick the corresponding 2D slice of the lookup table. Then, we shift it in the direction of β_{Li} and β_{Ri}, wrapping the values at the borders. This results in the negative log likelihood of each bin given a correspondence. By summing up negative log likelihood of each correspondence, we obtain the negative log likelihood over the discretized pose space. This is used to find the Maximum Likelihood solution (see Equation 3.40). Algorithm 1 summarizes this procedure.

Algorithm 1 Maximum likelihood estimator using a LUT

```
HIST ← 2D array of zeros
for all correspondences $\{\alpha_L,\beta_L,\alpha_R,\beta_R\}$ do
    compute $r=\frac{\tan(\alpha_R)}{\tan(\alpha_L)}$
    HIST += LUT(d($r$)) shifted by d($\beta_L$) and d($\beta_R$)
                where d($\cdot$) denotes discretization
end for
ML-solution ← bin2value(bin with highest value in LUT)
```

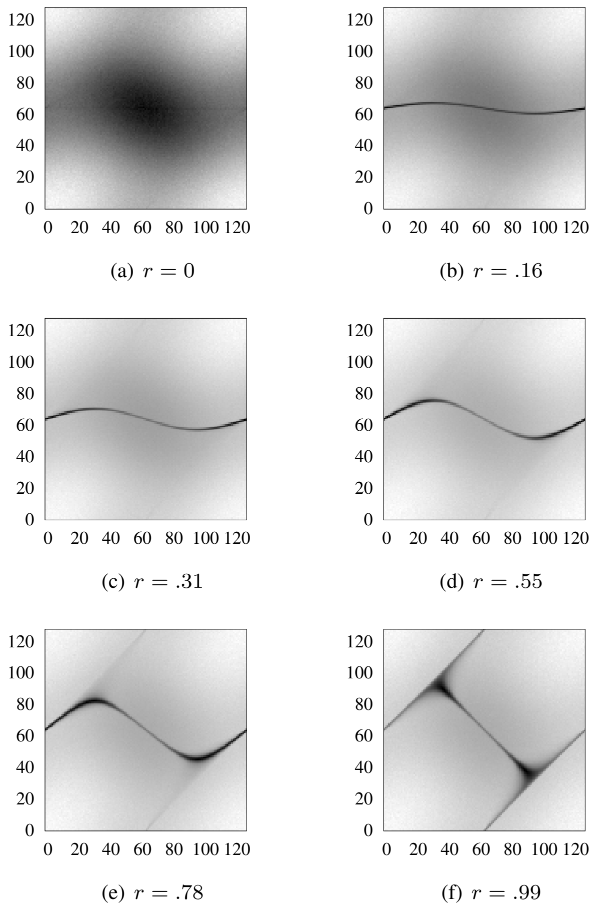


Figure 3.9: A few slices of a 128^3 bin lookup table representing the likelihood obtained from simulated data as denoted in Section 3.6.3. Again, white bin indicate low likelihood, while dark bins indicate high likelihood. This likelihood was also used throughout the experiments. The simulator and data used are described in Section 3.7. In (a) one can see that point correspondences with a r value close to zero, do not tell that much about the data. This is due to the fact that there is a high chance that it resulted from a mismatch. Note that the histogram corresponding to a r value close to 1 in (f) is almost symmetric. Indeed, if r is exactly 1 then $r = \frac{1}{r}$ and we could swap ϕ and θ.

3.6.3 Constructing a lookup table from data

The LUT can be constructed using an explicit error function. However, as we have seen, one can usually only approximate the true error using a first order approximation. Furthermore, it is difficult to model mismatches in the error function. We describe a different approach, which uses existing datasets of images correspondences with their ground truth poses. First, we explain how this can be achieved with a simulated model. Secondly, we show how to use real image data.

Simulated model

Sets of correspondences seen from different relative robot poses, can be generated using a simulator modeling the planar pose problem. When building such a simulator, one can use prior knowledge about the specific robot and the environment. In the following part, a minimalistic simulator is described which will also be used in the experiments.

The simulator consists of a set of randomly picked landmarks and random robot poses. For each iteration, a random point cloud of 3D landmarks is picked uniformly distributed inside a sphere of size 2 around the origin. For each run, two random camera poses on a circle with radius 1 in the x-y plane around the origin are chosen. From the two camera poses, the ground truth values for ϑ and ϕ is determined. Note that the distribution of ϑ and ϕ is approximately uniform.

A set of point correspondences is constructed by projecting some of the landmarks on a spherically shaped image surface with a radius of 1 around the camera pose. Thus, an ideal omnidirectional camera model is used with a full 360 degrees view angle in the horizontal and vertical direction.

An amount of normally distributed noise with zero mean and standard deviation 0.01 is added to these projections. This noise is not specified in the number of pixels, but in the space of the spherical image surface. For comparison, the value .01 corresponds to an angular error of approximately .57 degrees, which corresponds to about 6 pixels for a typical conventional mega pixel camera with a focal length of 8 cm. This amount of noise may seem quite large. However, it is not only used to model the noise of the imaging device itself, but also accounts for the simplifications of the camera model, the calibration errors and errors of the image key point extractors.

In addition, mismatches are added to the set of point correspondences by creating false correspondences between projections of different landmarks. We use a mismatch rate of 90%.

The end-result is a large set of relative poses (ϑ, ϕ) and associated correspondences ξ. From these $\frac{\tan(\alpha_R)}{\tan(\alpha_L)}$, $\vartheta - \beta_{Li}$ and $\phi - \beta_{Ri}$ are computed, and are put in a 3D histogram. Finally, each value is replaced by its negative log, resulting in a proper LUT.

The advantage of using a simulator is that we can easily generate large numbers of correspondences. In practice, the number of correspondences should be a few orders of magnitude larger than the number of bins in the LUT. Using the simple model described above we could generate $5 * 10^{10}$ samples in less than 10 hours, see Figure 3.9 for a visualization of this 3D lookup table with 128^3 bins.

Real image data

The drawback of using a simulator is that the used model will always only approximate real world data. Therefore, we now explain how real image data with ground truth relative pose data can be used. We assume that there is already a set of relative poses, each associated with a set of point correspondences extracted from the image pairs.

The main problem of using this data is that the relative poses are in general not uniformly distributed. This will cause the simplification proposed in Equation (3.55) to become invalid and moreover, will result in a bias in the LUT towards certain poses which are overrepresented in the dataset. However, we can easily compensate for this problem as follows.

When constructing the LUT, we explicitly take the probability of the relative pose into account (see Equation (3.54)). In a first step, a 2D discretized probability $p(\vartheta, \phi)$ is constructed by making a histogram for all poses in the dataset and normalizing it. Then, in a second step, the dataset is used to build the 3D LUT in the same manner as for the simulator, with the difference that for each pose correspondence it adds $\frac{1}{p(\vartheta,\phi)}$ to the 3D histogram. Again, each value of the histogram is replaced by its negative log, resulting in a proper LUT. Algorithm 2 summarizes the procedure to build a LUT. To keep the algorithm comprehensible, it ignores the symmetry between ϑ and ϕ.

A second problem, which cannot be circumvented, is that the amount of data in an image dataset is limited. As a consequence, we usually can not construct a LUT with a very large number of bins. In the next section, we evaluate, among other things, the consequences of such a smaller LUT.

Algorithm 2 Constructing a LUT from real data

```
PosePrior ← 2D array of zeros
for all training data $\{\vartheta, \phi, \alpha_L, \beta_L, \alpha_R, \beta_R\}$ do
    PosePrior(d($\vartheta$), d($\phi$)) += 1
end for
LUT ← 3D array of zeros
for all training data $\{\vartheta, \phi, \alpha_L, \beta_L, \alpha_R, \beta_R\}$ do
    compute $r = \frac{\tan(\alpha_R)}{\tan(\alpha_L)}$
    LUT(d($r$), d($\vartheta - \beta_L$), d($\phi - \beta_R$)) += 1/ PosePrior(d($\vartheta$), d($\phi$))
end for
LUT ← - log each entry of LUT
```

3.7 Experiments

The algorithms for planar relative pose estimation are evaluated using both simulated data, coming from a simple robot and camera model and real images from the datasets described in Appendix A. The focus is on comparing our Two-point algorithm with the Three-point and Eight-point algorithms, all in combination with RANSAC and M-Estimators and the histogram-based method. The methods are compared on the basis of their robustness against mismatches and noise.

3.7.1 Error measures

As is common in pose estimation literature, we evaluate the heading angle and the rotation angle (Nistér, 2004). The error of these estimated angles is computed by taking the absolute difference with the ground truth heading and rotation as in Stewénius et al. (2006). See Appendix A for a description of how ground truth data was obtained for the real home datasets. Because these error values are not normally distributed, mean errors and standard deviations are not that descriptive. Therefore, we mainly use the median to describe the location of the error distribution. In order to show the significance of the results, it is common to plot confidence intervals. To obtain these, every experiment is repeated 10 times with new data, each time determining a new median. A robust way to describe the spread of these medians is to use the Median of Absolute Deviations (MAD) which is given by:

$$\mathrm{MAD}(X) = \mathrm{median}_i \left(|x_i - \mathrm{median}_j(x_j)| \right). \tag{3.56}$$

Another important evaluation criterion is the computational time used by the algorithms. For all experiments, we report these times. However, we should note that evaluating the computational times of computer algorithms is difficult. It depends greatly on the way they are implemented and the computer architecture they run on. For these experiments, all algorithms were implemented in C++ and run on the single 2 Ghz CPU core of a Pentium PC.

3.7.2 Experiments on simulated data

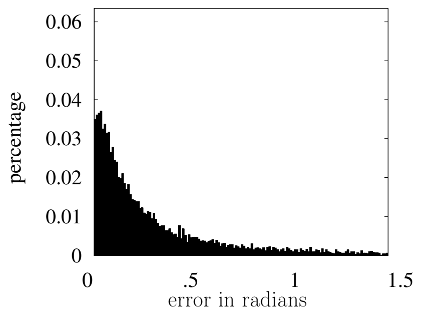
First, the different approaches are applied to simulated data, which allows us to control the projection noise and number of mismatches. It also allows us to add some non planar camera motion, simulating a ground floor that is not level or inclination of the robot.

Experimental setup

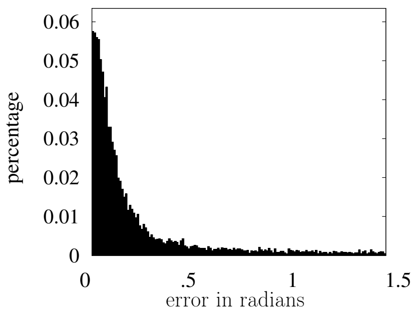
A lookup table was constructed using the simulator as described in Section 3.6.3. The number of bins to represent ϕ, ϑ and r were all 128, which caused the computational time to be comparable with that of the RANSAC + M-Estimator combined with the 3-point algorithm. The number of point correspondences used was 10^{10}, which took three hours to build. The error threshold for both RANSAC and the M-Estimator were set according to the projection noise of the simulator. Test data was picked using the same simulator as described in Section 3.6.3, which used a mismatch rate of 90%.

Resulting distributions

The distribution of errors is given in Figure 3.10. The accuracy of the histogram-based method is higher than that of the state of the art method. In addition, it is clearly visible that both distributions have long tails, showing the need of describing them using robust statistics.



(a) Heading error of 3pt RANSAC + M-Estimator.



(b) Heading error of LUT.

Figure 3.10: Comparison of the heading error of RANSAC + M-Estimator combined with the Planar Three-point algorithm and the proposed LUT ML method for 90% mismatches. The distribution of rotation errors shows a similar pattern.

Sensitivity to mismatches

To test the robustness of the different methods to mismatches we vary the number of mismatches, from 50% to 99%. In total, 10^5 iterations were conducted. In Figures 3.11(a) and 3.11(b) the resulting estimation errors are plotted.

Both the heading and rotation results show the same trend. The error of the RANSAC + M-Estimator with the 8-point algorithm, which does not take planarity into account, increases rapidly when more than half of the correspondences are mismatches. The RANSAC + M-Estimator with the 3-point is always better than the 2-point version. and 3-point algorithm behave similarly and start diverging at 65% mismatches. Note that the 3-point version is always slightly better than the 2-point version. The accuracy of the rotation estimates of the histogram-based estimator is higher than both the RANSAC + M-Estimator variants at more than 50% mismatches. For the heading estimates this is the case at more than 65% mismatch.

Sensitivity to violations of the planar assumption

In practice the motion of a robot is never strictly planar. Therefore, we test the behavior of the algorithms when small rotations with axes parallel to the ground floor are added to the poses. We built two separate LUTs for this experiment. The first one, denoted with “LUT-con” was created using the conventional simulator described in Section 3.6.3 using 10^7 simulated correspondences. The second one, denoted with “LUT-rot”, was also created with that simulator but in addition rotations were added about the two robot-axes parallel to the ground floor both up to .25 radians. This last LUT thus reflects the error resulting from slightly non planar motion.

In Figures 3.11(c) and 3.11(d), the median errors are plotted against the amount of pitch and roll, for a mismatch rate of 60%. As can be seen at an angle of .3 radians the combination of RANSAC and the Three-point algorithm, as well as the LUT-con method have the same or worse error as the combination of RANSAC with the Eight-point algorithm which is not influenced by non planar motion. Note that this corresponds to a percentage-grade of 31%, which is not realistic in a lot of robotic applications. It can be seen that the LUT-rot method, learned to be more robust against non planar motion.

3.7.3 Experiments on real data

The different approaches are also applied to real data, taken by an omnidirectional vision system in challenging home environments. The number of mismatches will vary substantially from almost 0% for consecutive image to 100% for images taken in different rooms.

Experimental setup

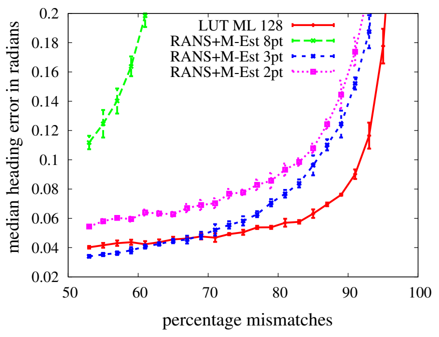
From every dataset described in Appendix A, we use every pair of camera images. It could however be that the pair of images was shot at the same position because the robot stood still and rotated on the spot. In that case, the heading cannot be determined. We discard these image pairs. Furthermore, if images are taken at more than 5 meters apart, then the chance of finding point correspondences is small. So we also discard these pairs. Still, for each set there are around 10^6 image pairs left.

To extract point correspondences from the image pairs, the SIFT algorithm is used (Lowe, 2004). First, omnidirectional images are mapped to panoramic images (Bunschoten, 2003), from which the SIFT feature points are found. These features are described by the standard SIFT descriptor of 128 dimensions. If two features in the same image have a small distance in descriptor space then they are removed. A set of point correspondences between two images is determined by applying the standard matching scheme as described in Lowe (2004). This resulted in on average 25 matches per image pair. The groundtruth relative pose was computed from the groundtruth robot positions.

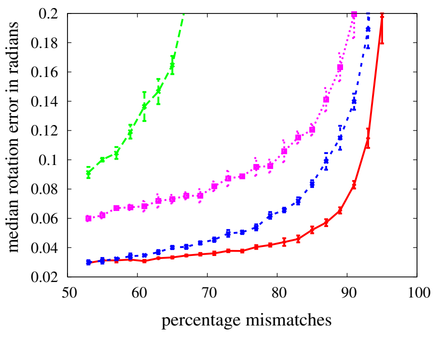
Sensitivity to mismatches, trained with simulated data

In order to evaluate the performance of the methods, we would like to vary the number of mismatches. This cannot be controlled in real data, therefore we made subsets of the data on the basis of the distance between the poses. We assume that for larger distances it is more difficult to find matching features. For the Histogram method we first use a LUT constructed using the simulator described in Section 3.6.3 and see how well it compares to state of the art methods. In Figure 3.12(a) and 3.12(b), the heading and rotation error of the different methods is plotted as a function of the distance between the images for the Almere 4 dataset . It is clear that on a whole the errors are much larger than was the case for the simulation data. This is partly due to the fact that some of the views were obstructed by furniture, walls or people walking in the environment.

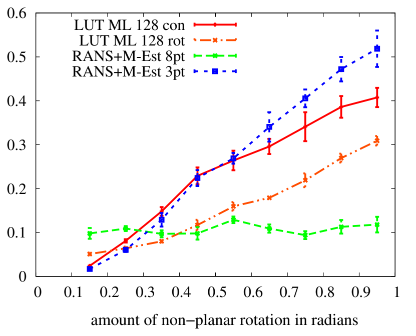
In the plot of the heading error (Figure 3.12(a)), one can see that the RANSAC + M-Estimator combined with the Two-point algorithm is outperformed by the Three-point algorithm version, which in turn is clearly outperformed by the Histogram method for distances larger than 1.5 meters. The accuracy of the RANSAC + M-Estimator combined



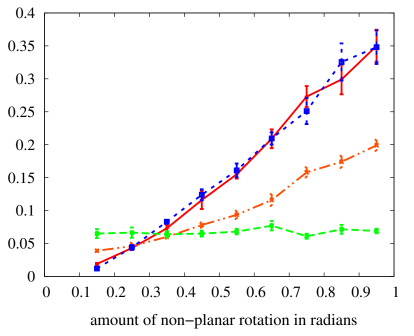
(a) Heading errors, simulation data



(b) Rotation errors, simulation data

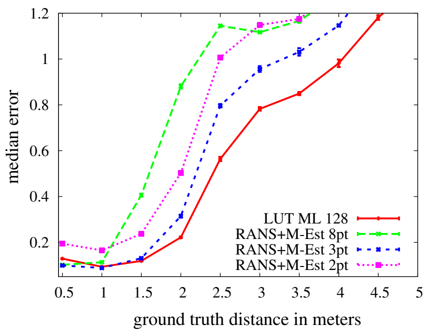


(c) Heading errors, non planar simulated data

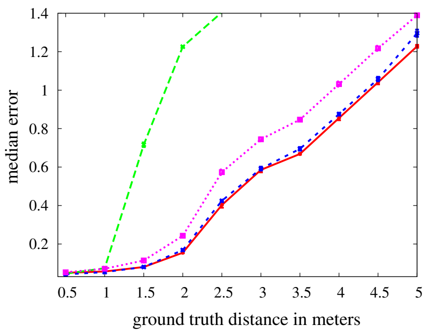


(d) Rotation errors, non planar simulated data

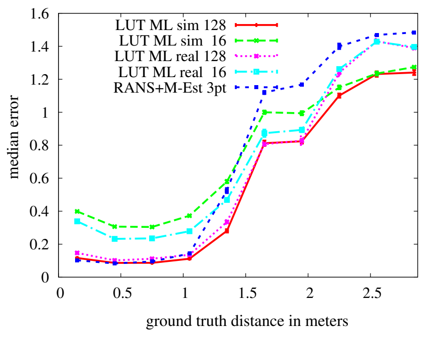
Figure 3.11: Comparison of the different relative pose estimators using simulated data. (a) and (b) show the heading and rotation errors for different numbers of mismatches. (c) and (d) show the heading and rotation errors for different amounts of non-planar motion. The MAD statistic is used to draw confidence intervals of the medians.



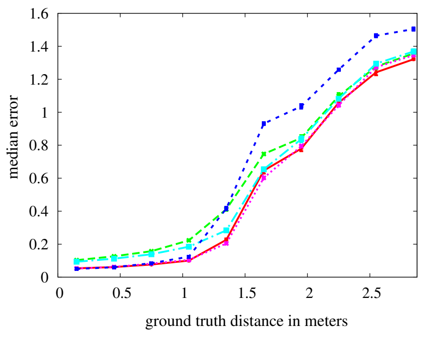
(a) Heading errors, Almere 4



(b) Rotation errors, Almere 4



(c) Heading errors, Spaan 1



(d) Rotation errors, Spaan 1

Figure 3.12: Comparison of the different relative pose estimators using real images. In 3.12(a) and 3.12(b) the errors are shown for the Almere 4 dataset. For this test the LUT was built using a simulator. In 3.12(c) and 3.12(d) the errors are shown for the Spaan 1 dataset. For this test the LUT was built using images from the Almere 4 dataset. The MAD statistic is used to draw confidence intervals of the medians. However, these intervals are relatively small, and hard to see in the graphs.

Table 3.1: Average computational time usage per relative pose estimate in milliseconds for the different methods.

Hist ML				RANS+M-Est		
128^3	64^3	32^3	16^3	8pt	3pt	2pt
1.3	0.28	0.07	0.036	3.6	3.8	0.68

with the Eight-point algorithm is not that bad as compared to the methods taking the planarity constraint into account. This could have been caused by the fact that the robot was leaning over when accelerating, slightly violating the constraint. For the rotation error (Figure 3.12(b)), the improvement of the Histogram method over the RANSAC + M-Estimator with Three-point method is less pronounced.

Sensitivity to mismatches, ingtrained with real data

Next, we constructed a lookup table using all the image pairs of the Almere 4 set that were within a 5 meter distance and applied it to the Spaan 1 set. We used two different bin sizes, the first had 128^3 bins and the other 16^3. The Maximum Likelihood solutions based on these two tables were compared to the RANSAC + M-Estimator combined with the Three-point algorithm and solutions given two LUTs based on the simulator, also with dimensions 128^3 and 16^3.

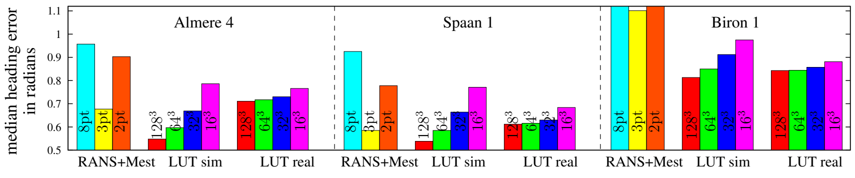
In Figure 3.12(c) and 3.12(d), the results are shown. As can be seen, the overall accuracy is less than for the Almere 4. A reason for this could be the motion blur, caused by the bad illumination of this dataset. The histogram with 128^3 bins based on the simulator performs best, followed by the 128^3 bins LUT-based on the real images. Probably this is due to the limited number of point correspondences in the real image set. For the much smaller LUTs with 16^3 bins, this seems to be less problematic, visible by the improvement of the LUT-based on real images over the one based on simulated data.

Averages over the data sets

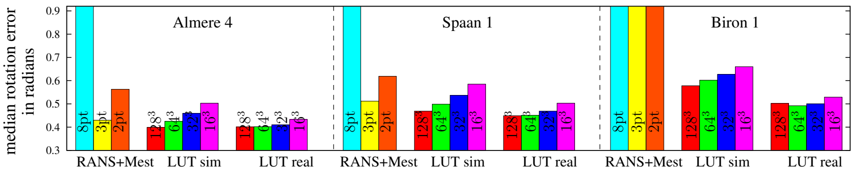
Application on other datasets and different bin sizes resulted in comparable errors. Figure 3.13 summarizes these results. Note that all three RANSAC-based methods failed to robustly estimate poses for the difficult Biron 1 set in contrast to the proposed method.

Bin size vs CPU time

To evaluate the influence of different bin sizes for the lookup table, we tested the Histogram method for different numbers of bins. In Table 3.1, the average computational time in milliseconds is given per image pair for the Almere 4 set (other datasets showed similar trends). As can be seen, small lookup tables result in a large speed-up, but this comes at the cost of a larger error (see Figure 3.12). The RANSAC + M-Estimator combined with the Planar three point algorithm is three times slower than the Histogram method with 128^3 bins.



(a) Heading errors



(b) Rotation errors

Figure 3.13: (a) Heading errors and (b) rotation errors for the different estimators applied to all three datasets. The MAD-based confidence intervals are omitted, because they were very small and hard to see.

3.8 Discussion

An important advantage of the histogram-based pose estimator is that it provides a full likelihood over the discretized space of possible relative poses. Thus, apart from computing a Maximum Likelihood solution as shown in the Experiments section, this could make the method useful for a range of other applications.

The proposed method could be readily applied to particle filter-based robot localization schemes (Gross and Koenig, 2004), where each hypothesized robot pose can be weighed by the likelihood given newly acquired images. On top of this, geometric SLAM could benefit from the proposed method. If needed, a parametric uncertainty model of the Maximum Likelihood can be estimated easily from the full likelihood. This could, for example, be achieved by fitting a Von Mises or mixture of Von Mises distributions on the discretized likelihood space (Bishop, 2006).

Another task that is very much suited for the proposed example is that of topological mapping. Some state of the art topological mapping approaches use proper probabilistic data association techniques to compare pairs of images (Cummins and Newman, 2009). However, in addition they apply ad hoc rules to check whether the matched point correspondence fit in a certain local geometry by computing the relative pose. Because of the probabilistic nature of the proposed method, it is straightforward to combine it with these proper data association techniques, ending up in a fully probabilistic topological mapping method.

3.9 Conclusion

In this chapter we have investigated the planar relative pose problem thoroughly. By doing so, we have provided some new insights and developed some powerful algorithms. The most interesting insight is that two point correspondences do not always define a single planar relative pose, but half of the time define two relative poses. We developed a novel algorithm that computes the single or double solution in closed form from two point correspondences.

We continued by developing algorithms for the planar pose problem that can cope with noisy data including large percentages of mismatches. This resulted in a combination of existing robust estimators, including RANSAC and the M-Estimator, with known error functions adapted for the planar pose problem. Furthermore, we have shown the advantage of discretizing and analyzing the complete solution space. In the planar motion case this is two dimensional. Probabilistic methods were proposed that learn the likelihood over this space from a training set of representative images. Experiments on challenging image sets acquired in real homes showed a 20% increase in accuracy with respect to state of the art methods consisting of a planar constrained RANSAC and M-Estimators.

In addition, an efficient technique was presented for building a concise lookup table of the likelihood. This reduces the estimation process to simple lookups. Computing a full likelihood given two images costs as little as 36 microseconds, in comparison to the 3 milliseconds RANSAC uses. This could even be improved, for example, by using a multi-resolution approach as described in Olson (2009a), in which a small lookup table is used to isolate candidate areas for the ML solution. These are then investigated further using a bigger lookup table. Another possibility is implementing the method on a GPU, which can much more quickly manipulate 2D histograms.

4 Image similarity for view-based mapping

In Chapter 2, we have seen that state of the art image similarity measures based on local image features often use local geometric constraints to discard falsely matched features. Implicitly, this results in a measure that reflects how well relative pose information can be extracted from point correspondences. In the previous chapter, we developed a new algorithm to estimate the relative pose. In this chapter, we propose a new image similarity measure based on this pose estimation method. Because the developed pose estimation method of Chapter 3 determines a full likelihood over the complete space of possible relative poses given the point correspondences, we can also compute the probability of the most likely relative pose. This probability is used to define a measure of image similarity. We compare it with other state of the art similarity measures: a simple approach based on feature matches, a RANSAC-based approach and the Bag Of Words approach FABMAP.

4.1 Introduction

In most, if not all, view-based mapping systems it is necessary to determine whether two images partly depict the same structures of the environment (Eustice, 2005; Torralba et al., 2003; Konolige and Bowman, 2009). An image similarity function that determines the similarity of two given images is the basic tool to perform this task (Datta et al., 2008). The quality of the resulting view-based maps depends for a large part on the robustness of this function.

Popular similarity functions used in view-based mapping systems base the image similarity on the number of point correspondences found. In order to discard falsely matched image points, it is common to first estimate a relative pose from the correspondences using the robust RANSAC algorithm and then discard correspondences that do not fit this pose (Eustice, 2005; Newman et al., 2006; Fraundorfer et al., 2007; Konolige et al., 2009). In the related semantic place recognition task, this same technique is commonly used to measure the similarity betweeen images of buildings and touristic sites (Segvic et al., 2009; Chum and Matas, 2010; Li et al., 2008; Philbin et al., 2007).

In the previous chapter, we already discussed two problems with RANSAC. First, it requires an explicit error model that describes the noise characteristics of the vision sensor. Second, it makes an explicit distinction between inliers and outliers using a hard threshold. An additional problem, which we will explain in detail in this chapter, is that counting the number of inliers does not give a proper measure of image similarity. It is not a probabilistic measure, making it inappropriate to combine with other probabilistic methods, such as for example the FABMAP method (Cummins and Newman, 2008).

In Chapter 3 we have developed a completely different method for determining the relative robot pose, given that the robot moves over a planar ground floor with a rigidly mounted vision system. The method learns the noise characteristics in the form of a non parametric likelihood using a set of training images and their ground truth poses. We have shown that the resulting most likely relative poses are more accurate than the poses determined by state of the art methods. Because it estimates the full likelihood over the space of all relative poses, we also have access to a full density estimate[1] of the relative pose given the correspondences.

In this chapter, we use this full density estimate of the estimated pose as a measure of image similarity and compare it with the state of the art RANSAC-based method and the FABMAP 2.0 method. As in the previous chapter, image sets of the home environments are used. In Section 4.2, we describe both the state of the art RANSAC-based method as well as our new algorithm. To compare the algorithms, we have to establish a way of evaluating the image similarity results. In Section 4.3, we review existing evaluation criteria and define the two criteria we use in the experiments. In Section 4.4, we evaluate how well the similarity measures perform in a semantic place recognition setting. We conclude the chapter with a discussion in Section 4.5 and a conclusion in Section 4.6.

4.2 Image similarity measure

In this section, we first give a definition for the popular RANSAC-based image similarity method. We will see that this method has some fundamental problems. To overcome these, we propose a similarity measure based on the novel pose estimation method based on lookup tables, described in the previous chapter.

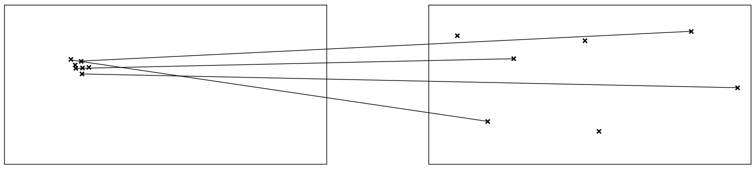
4.2.1 RANSAC-based similarity measure

The RANSAC-based similarity measure works as follows. First, invariant local image features are extracted from the images i and j, for example using the SIFT or SURF method. These are matched to find a set of n_{ij} local point correspondences $\{\xi_1, \ldots, \xi_{n_{ij}}\}$. Given these correspondences, we can already define a simple feature-based similarity measure (as used in for example (Andreasson et al., 2008)):

$$S_{ij}^F = \frac{n_{ij}}{\frac{1}{2}(F_i + F_j)}, \tag{4.1}$$

where F_i and F_j are the numbers of features found in respectively image i and image j. This measure can be improved by applying geometric constraints on the correspondences as follows. From the correspondences, a relative camera pose $(\vartheta^R_{\text{ML}}, \phi^R_{\text{ML}})$ is estimated using a combination of least square estimators and robust methods (as explained in the previous chapter in Section 3.5). The most important step in this estimation process is discarding the usually large number of outliers which are mostly caused by mismatched

[1]Because we discretize the solution space in a finite number of bins, it is perhaps better to use the term *probability mass function*. We will however use the term density, because the underlying variables are continuous.


(a) Two images with four feature matches

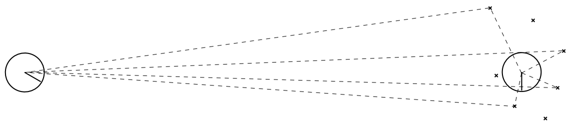

(b) Possible robot poses causing the matches

Figure 4.1: A visualization of a degenerate formation of point correspondences between two images (a) and a possible relative camera pose given these correspondences (b).

image features. This step is generally performed using the RANSAC method, which was described in Section 3.5.2.

The similarity between the two images is based on the number of point correspondences that fit the resulting estimated relative camera pose. Whether a correspondence ξ fits a relative pose is determined using a distance measure $d\left(\xi, (\vartheta^R_{\text{ML}}, \phi^R_{\text{ML}})\right)$. This commonly approximates the reprojection error of the correspondence given the relative camera pose (see for example (Hartley and Zisserman, 2003)). In our experiments, we will use the Sampson distance as given in Section 3.5.1. If this distance is below a preset threshold c, then it is said to fit the relative pose. The final similarity value is determined by dividing the number of fitting correspondences with the average number of features found in both images as in Equation (4.1). To summarize, the similarity value S^R_{ij} between image i and image j is defined as follows:

$$S^R_{ij} = \frac{1}{\frac{1}{2}(F_i + F_j)} \sum_k H\left(d\left(\xi_k, (\vartheta^R_{\text{ML}}, \phi^R_{\text{ML}})\right) - c\right), \tag{4.2}$$

where $H(\cdot)$ denotes the Heaviside step function.

The rationale of this similarity function is that it determines the ratio of the number landmarks in the scene that can be seen in both images with respect to the total number of landmarks in the images. Let us look more closely how this method works out in practice.

An explicit distinction is made between correspondences that resulted from the same landmark depicted in the two images and the correspondences that were caused by, so-called, mismatches. This distinction is made on the basis of the pose estimation result, and therefore depends on the success of the pose estimation procedure. It is known, however, that this pose estimation sometimes fails. A common problem is that the correspondences can lie in a degenerate formation.

Figure 4.1 visualizes such a degenerate formation. In Figure 4.1(a) two images are visualized. In the image on the left all features are close to each other. In the right image, however, the features are spread more or less uniformly over the image plane. In Figure 4.1(b) a relative camera pose is visualized that can explain this set of point correspondences. The camera equipped robot on the right is surrounded by landmarks, while the robot on the left is standing far away seeing all landmarks at more or less the same angle. Actually, it does not matter how the features are distributed in the right image. They will approximately fit the visualized camera pose and most of them are counted as correct matches. This causes the similarity value to be artificially high with respect to the similarity values of sets of point correspondences which do not lie in a degenerate formation.

Such degeneracies can actually be detected. In the example of Figure 4.1, it does not matter where the left robot is positioned, as long as it is far from the right robot position. Thus, there are multiple camera poses for which the correspondences fit. The estimated pose is actually not that distinctive. The pose estimator indeed tried to find the relative pose that was most likely relative camera pose. So, how can a similarity function take into account whether the estimated relative camera pose is rather distinctive or not at all distinctive?

This can be answered easily using some basic probability theory. We should divide the likelihood of the point correspondences, given the estimated relative camera pose, by the sum of the likelihoods given all possible poses. The problem with the RANSAC-based similarity method is that it only provides the likelihood for a specific relative camera pose and thus this normalization cannot be performed.

4.2.2 LUT-based similarity measure

In the previous chapter, we have developed a new method to estimate the likelihood of the relative camera pose given a set of point correspondences. By discretizing the space of all possible relative poses and efficiently computing the likelihood of each pose, we obtained an estimate of the likelihood over the complete space of poses. We used this discretized density function to estimate the relative camera pose that was most likely by selecting the pose with the maximum likelihood:

$$(\vartheta^L_{\text{ML}}, \phi^L_{\text{ML}}) \quad = \quad \underset{(\vartheta,\phi)}{\arg\max}\, p(\xi_1, \ldots, \xi_n | \vartheta, \phi). \tag{4.3}$$

In this chapter, the full distribution is used to normalize the probability of the maximum likelihood. This is done by applying Bayes' rule. In this way, we obtain a similarity measure which takes into account degenerate formations of the set of point correspondences:

$$\begin{aligned} S^L_{ij} &= p(\vartheta^L_{\text{ML}}, \phi^L_{\text{ML}} | \xi_1, ..., \xi_n) \\ &= \frac{p(\xi_1, ..., \xi_n | \vartheta^L_{\text{ML}}, \phi^L_{\text{ML}}) p(\vartheta^L_{\text{ML}}, \phi^L_{\text{ML}})}{p(\xi_1, ..., \xi_n)} \\ &= \frac{p(\xi_1, ..., \xi_n | \vartheta^L_{\text{ML}}, \phi^L_{\text{ML}}) p(\vartheta^L_{\text{ML}}, \phi^L_{\text{ML}})}{\sum_{\vartheta,\phi} p(\xi_1, ..., \xi_n | \vartheta, \phi) p(\vartheta, \phi)}, \end{aligned} \tag{4.4}$$

where we use the sum rather than the integral to indicate that the estimated likelihood is represented by a discretized density function. In our case, the prior relative pose $p(\vartheta, \phi)$

is taken to be uniform, as is the case for the estimated full likelihood. Thus Equation (4.4) can be simplified :

$$S_{ij}^{L} = \frac{p(\xi_1, ..., \xi_n | \vartheta_{\mathrm{ML}}^{L}, \phi_{\mathrm{ML}}^{L})}{\sum_{\vartheta,\phi} p(\xi_1, ..., \xi_n | \vartheta, \phi)}. \tag{4.5}$$

This similarity measure does not only solve the degeneracy problem. It also gives a more appropriate measure for view-based mapping applications. Namely, it measures the probability that we can estimate the local relative geometry between the camera poses given two images. This is clearly relevant for applications, such as view-based SLAM and view-based robot navigation. In this chapter, however, we focus on the task of topological mapping which is not directly related to pose estimation. In the experiments in Section 4.4, we compare it with the RANSAC-based measure. However, first we have to define a proper evaluation scheme.

4.3 Evaluation criteria

Before applying the described similarity measures, we have to define useful evaluation criteria for our application. In this section, we first describe the current practice with respect to evaluating view-based topological mapping methods. We then describe the two evaluation criteria we use in the experiments.

4.3.1 Related work

In the related semantic place recognition task it is common to evaluate methods quantatively. The common scheme, as used in most content-based image retrieval applications, is to assign a single semantic label - such as “kitchen” or “Paris” - to each of the images (Torralba et al., 2003). This assignment is usually done by hand (Russell et al., 2008), sometimes aided with additional information such as GPS data or the sequential order of the images (Zheng et al., 2009; Zivkovic et al., 2008). Image pairs with the same label are then regarded as being similar. This same approach is often used to evaluate view-based topological mapping methods (Ulrich and Nourbakhsh, 2000; Pronobis et al., 2010; Kosecká et al., 2005). However, as discussed in Chapter 1, there is a mismatch between these tasks.

A rigorous quantitative evaluation method for the task of view-based topological mapping itself does not exist. Commonly, ground truth positioning data is used to evaluate image pairs with a high similarity value. For example, Valgren and Lilienthal (2010) describe how images taken more than 10 meters apart should have a similarity value below a certain threshold. In Cummins and Newman (2009) this distance is set to 40 meters. Such a measure only takes account of the distance between the camera poses and disregards other properties, such as viewing direction, walls blocking the view of the camera or the average distance to visual features.

In Cummins and Newman (2008) ground truth similarity was determined by checking image overlap by hand. This was achievable for the particular image set used in their evaluation, because the camera was looking perpendicular to the driving direction and the viewed scene was at more or less the same distance throughout the robot trajectory. In general, however, such an approach would lead to subjective evaluation results.

In Frese and Neira (2009) a dataset is proposed which includes hand-labeled associations of artificial white circles placed on the ground and seen in the images. This data was used for example in Olson (2009b) to evaluate a view-based mapping procedure. A drawback of this dataset is that the circular objects are all put on the same plane, namely the ground floor. Because of this, the resulting sets of correspondences actually are degenerate cases (Kanatani, 1996).

In the field of image registration, which is commonly used for fusing medical images taken from the same patient, a similar kind of evaluation has to be performed. However, in this field researchers resort to hand-labeled data through the means of control points to evaluate different methods (Zitova and Flusser, 2003). In order to keep the experiments objective and reproducible, we will abstain from using subjective hand-labeled data.

4.3.2 Our evaluation measures

In our application, we use the image similarity measure to find links in a topological view-based map. In Chapter 1, we explained that such a link indicates that two images depict an overlapping part of the environment. Thus, we should evaluate if the found image similarity values are indeed high if such an overlap exists, and low if it is not. Note that some non-overlapping parts of an environment could be visually indistinctive from each other. In such a theoretic case, even the perfect image similarity measure cannot reach 100% accuracy. However, we consider only practical situations and real image similarity measures.

The problem is that the concept of depicting an overlapping part is ill defined. Specifically, in the home environments which are the focus of this thesis, we might assume that different rooms do not contain any overlap. However, in the datasets used in our experiments, the doors of the rooms were always open. Images taken in a certain room could therefore also depict small parts of other rooms in the house. The only case for which we can be sure that no overlap exists, is if the two images were taken in different homes. This evaluation criterion is used in experiments in Section 4.4. This evaluation measure is very similar to the standard labeled place recognition method, with the difference that we avoid ambiguities in the labeling process.

A drawback of this evaluation measure is that links found between nodes in the same house are always regarded as correct, even if they were based on only false point correspondences and could not have resulted in a reliable pose estimate. We can formulate a stricter evaluation criterion of a view-based link, by using the fact that the similarity values should give an indication of the success of the pose estimation.

We can easily evaluate the success of the pose estimation method, by comparing the estimated poses with the ground truth poses, which are available for the home datasets. Given a certain error bound we can determine whether the pose was successfully estimated or not. In the experiments, we only apply a bound on the estimated rotation and not the estimated heading. We have done this because the ground-truth rotations are more reliable than the ground-truth headings.

Using this stricter evaluation measure we can compare the RANSAC-based similarity measure with our proposed method. The stricter evaluation criterion can falsify links between images that were taken in the same home. Nevertheless, note that it does not falsify all false links. By accident an estimated pose given two images could be close to the ground truth, even if based on only false feature matches.

4.4 Experiments

In these experiments, we evaluate different similarity measures in a place recognition setting. The similarity measures should find a high similarity value for pairs of images that were taken in the same home and a low similarity value for pairs of images taken in two different homes. This allows us to compare the discussed similarity measures based on pose estimation, namely the RANSAC-based method and the LUT-based method, with other measures proposed specifically for solving this task, namely a simple feature-based method without geometric constraints and FABMAP 2.0. Thus, besides evaluating the new LUT-based method, the experiment evaluates if measures which are actually based on the ability to estimate the relative camera are indeed good similarity measures for solving the task of place recognition, as claimed in Chapter 2. In addition we evaluate if the similarity measures that use relative pose estimation, find a high similarity value for image pairs for which the estimated pose is indeed correct.

4.4.1 Evaluation measures

The results of the experiments will be presented and evaluated in a couple of ways.

In topological mapping literature it is common to visualize the results as connectivity graphs, using the ground truth camera positions to place the different nodes (Konolige et al., 2009). Each image pair with a similarity value above a chosen threshold, is plotted as a link in a graph connecting the corresponding two nodes. In our case the positions of the camera in a certain home is accurately known. Yet relative positions between different homes are not defined (the distance between Amsterdam and Bielefeld is not measured). Therefore, we artificially position nodes from different places far apart.

A more concise presentation, which is also used in place recognition literature (Ranganathan and Dellaert, 2007), is the confusion matrix. In our case, this is a symmetric square matrix where the rows and columns denote the different homes. Each element is filled with the total number of links from the home corresponding to its row to the home corresponding to its column. The main diagonal of this matrix represents the correctly matched image pairs per place; the true positives (TP^{home}). The off-diagonal entries represent the matched image pairs of different places; the false positives (FP^{home}).

For both the confusion matrices and the graphs we should choose a threshold for each similarity measure. A fair way to do this, is to choose it in such a manner that the number of matches is equal among the different measures.

A way to present the results independently from the chosen threshold, is using ROC curves (Receiver Operator Characteristic) that express the recall against the false alarm rate by varying the threshold value. The recall for the home recognition evaluation is defined as

$$\text{recall} = \frac{TP^{\text{home}}}{P^{\text{home}}}, \tag{4.6}$$

with the number of positives P^{home} defined as the total number of correct image pairs in the dataset:

$$P^{\text{home}} = \sum_{\text{label}} n_{\text{label}}(n_{\text{label}} - 1)/2, \tag{4.7}$$

where n_{label} is the number of images taken in the same room "label". Note that we regard all image pairs taken in the same house as positives, although part of them have no visual overlap. This objective measure will therefore result in a relatively low ROC as compared to other studies where a stricter set of positives was hand-picked (Valgren and Lilienthal, 2010; Cummins and Newman, 2009).

The false alarm rate for the home recognition evaluation is defined as:

$$\text{false alarm rate} = \frac{FP^{\text{home}}}{N^{\text{home}}} \tag{4.8}$$

with the number of negatives N^{home} defined as the total number of incorrect image pairs in the dataset:

$$N^{\text{home}} = n_{\text{total}}(n_{\text{total}} - 1)/2 - P^{\text{home}}, \tag{4.9}$$

with n_{total} the total number of images.

A similarity measure with an overall higher ROC curve thus results in more correct matches and fewer incorrect ones. For most robot applications, it is also interesting to look at the onset of the ROC curve. The onset corresponds to a false alarm rate of zero, which is preferable for most mapping applications. Wrong links are commonly catastrophic.

For the RANSAC and the LUT methods, additional ROC curves are computed using the more strict evaluation measure by determining if the pose estimation was successful. An estimated pose is deemed correct if the estimated rotation is within .5 radians of the ground truth rotation. For the total number of negatives and positives, the same values are used as for the home recognition based ROC curves.

Finally, as in the previous chapter, the computational cost of the methods is an important evaluation measure.

4.4.2 Data

Like in the previous chapter, we use the images acquired in the real homes (see Appendix A). The three image sets are joined to form a set of 5147 images in total. From these we randomly picked a set of $13 * 10^5$ image pairs, 10% of all possible image pairs. Of these image pairs, around $5 * 10^5$ are from the same home and should thus get a high similarity value and $8 * 10^5$ are from two different homes and should thus get a low similarity value.

From these image pairs, sets of point correspondences are created in a similar fashion as described in Section 3.7.3. In order to cope with the large number of image pairs, the images were compared using SURF features with a small descriptor of only 32 values. These are more efficient and require less storage capacity than SIFT features.

4.4.3 Similarity measures

The following similarity measures are compared:

Features A simple feature-based measure, in which the number of point correspondences found by comparing the SURF features is normalized by the average number of features found in the two images (see Equation (4.1)). This method does not apply relative pose constraints.

RANSAC The state of the art method as described in Section 4.2.2 and 3.5, based on the number of inlying SURF features which uses a combination of the RANSAC algorithm, an M-estimator and the Planar Three-point algorithm. This number is again normalized by the average number of features (see Equation (4.2)).

LUT128 The novel similarity measure proposed in Section 4.2.2, based on the discretized full likelihood estimate of Section 3.6, with a lookup table of 128^3 bins (see Equation (4.5)). The lookup table was constructed using all the image pairs of the Almere 1 dataset, which was taken in the same home as the Almere 4 dataset, yet following a different path and under lower dynamic conditions.

LUT16 The same as LUT128, but with a lookup table of 16^3 bins.

FABMAP The mapping method FABMAP 2.0, described in Cummins and Newman (2009) that does not use relative pose constraints. We should note that FABMAP is actually more than an image similarity measure, because it uses some additional information based on the sequential nature of the images (as discussed in Section 2.3.2). In addition, it assumes that sequential images are non overlapping. Therefore, we used a subset of the image set, as advised in the FABMAP software documentation, using only every 10th image. This results in a total of 515 images, which in turn resulted in about $13 * 10^4$ image pairs.

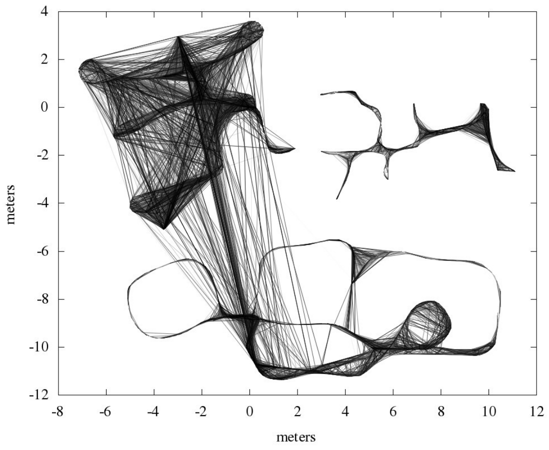
4.4.4 Results: connectivity graphs and confusion matrices

In Figures 4.2 to 4.4, the connectivity graphs - resulting from the different similarity measures - are plotted. In the top left of the graph, on finds the links associated with Biron 1 set. In the top right, one sees the links associated with the Spaan 1 set and at the bottom those of the Almere 4 set. The nodes of the graph, which denote the images, are not plotted to avoid clutter in the figures. For each graph, except the one in Figure 4.4, we plotted the 30,000 links with the highest similarity values. The color intensity of the plotted link indicates the height of the similarity value.

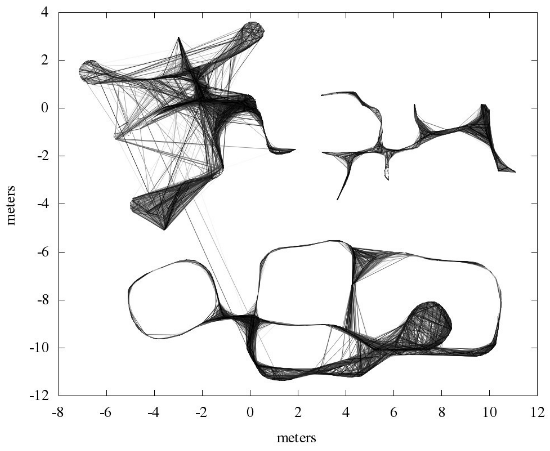
Most of the measures result in properly looking connectivity graphs, where a lot of links are found between images that were geometrically close to each other. Table 4.1 gives the confusion matrices associated with the connectivity graphs.

The Features method (Figure 4.2(a)) resulted in 179 wrong links between the Biron 1 and Almere 4 set. Most of the falsely linked images of Almere 4 were taken in a small hallway, which had few visual features like the Biron home. The RANSAC-based graph (Figure 4.2(b)) also has 3 of these bad links. In addition, some of the linked images within the Biron 1 set are from non-adjacent rooms and thus, most probably, based solely on mismatched image features.

The LUT128-based graph (Figure 4.3(a)) does not link images of different homes. Furthermore, there seem to be few links between non-adjacent rooms. The LUT16-based graph (Figure 4.3(a)) looks very similar but has a single erroneous link.

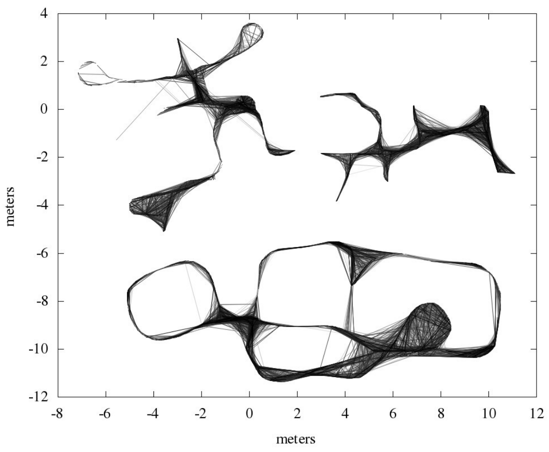


(a) Features

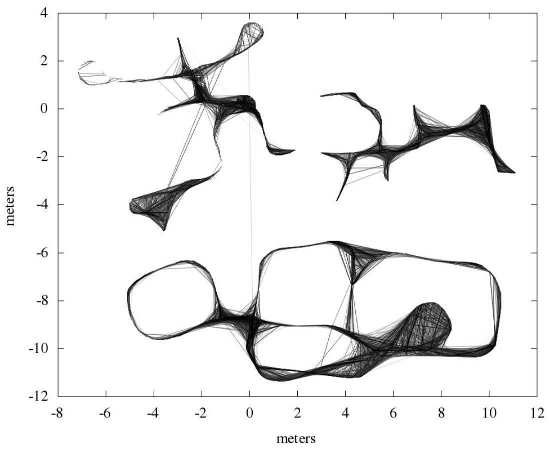


(b) RANSAC

Figure 4.2: Connectivity graphs of 30,000 links, based on (a) the purely Feature-based similarity measure and (b) the state of the art RANSAC-based similarity measure.



(a) LUT128



(b) LUT16

Figure 4.3: Connectivity graphs of 30,000 links, based on the two LUT-based similarity measures using lookup tables of (a) 128^3 bins and (b) 16^3 bins.

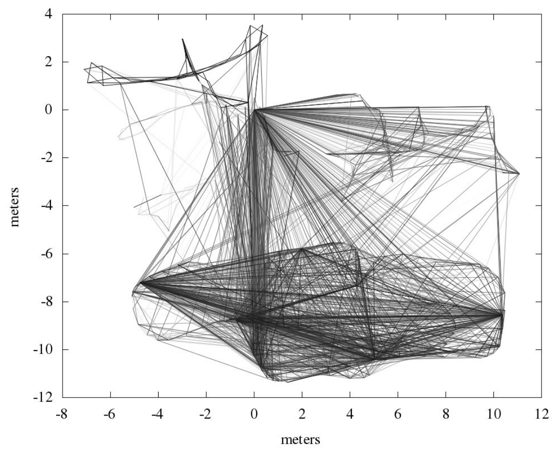


Figure 4.4: The connectivity graph of 3,000 links based, on the FABMAP similarity measure.

Table 4.1: Confusion matrices for the different similarity measures of the top 30.000 image pairs.

		Almere4	Spaan1	Biron1
Features	Almere4	11432	0	179
	Spaan1	0	5866	0
	Biron1	179	0	12523
RANSAC	Almere4	12630	0	3
	Spaan1	0	6375	0
	Biron1	3	0	10992
LUT128	Almere4	14350	0	0
	Spaan1	0	8840	0
	Biron1	0	0	6810
LUT16	Almere4	14313	0	1
	Spaan1	0	8992	0
	Biron1	1	0	6694
FABMAP (3000 pairs)	Almere4	1790	40	328
	Spaan1	40	306	143
	Biron1	328	143	393

Table 4.2: The number of correctly found image pairs at zero false rate of the different methods, corresponding to the onset of the ROC curves. (*) Note that the FABMAP method was applied to only 10% of the image pairs and thus this result should be multiplied by 10 when compared with the other methods.

Method	Features	RANSAC	LUT128	LUT16	FABMAP
# error free	16377	26402	30189	27314	430*

As described in Section 4.4.3, the FABMAP method was applied to considerably fewer images, resulting in a 10th of the number of evaluated image pairs. For fair comparison, only 3,000 links are plotted (Figure 4.4). These, however, include a lot of false links. Looking more closely, it appears that most of these false links originate from only a few images. Inspection of the position data revealed that the robot was moving relatively slowly, while taking these particular images. This might have influenced the results.

4.4.5 Results: ROC curves given home recognition

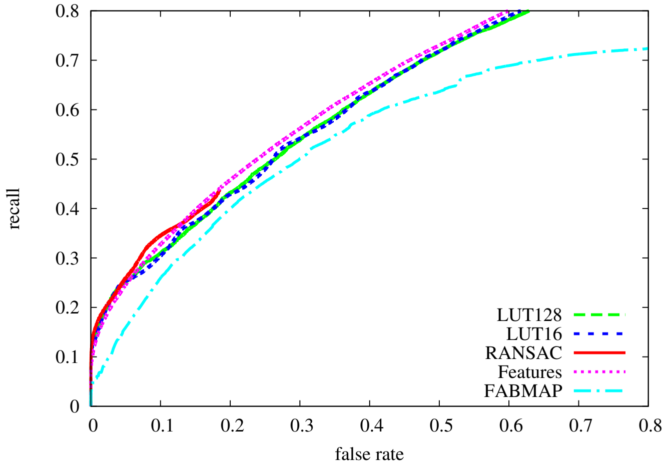
Figure 4.5 shows the ROC curves of the different methods, based on the home recognition evaluation. The complete ROC curves (Figure 4.5(a)) show that in general there is not much difference between the performance of the different similarity measures, except for the FABMAP method, which has a considerably lower performance. For most mapping applications, however, it is more interesting to see the behavior of the measures at low false rates.

Figure 4.5(b) shows the first 10% of the recall, which corresponds to about $5 * 10^4$ correct image pairs, and 0.03% of the false rate, which corresponds to about 240 image pairs between different homes. It can be seen that the RANSAC and the LUT measures have more or less the same performance with a slight advantage for RANSAC. The Features and FABMAP methods are clearly less accurate.

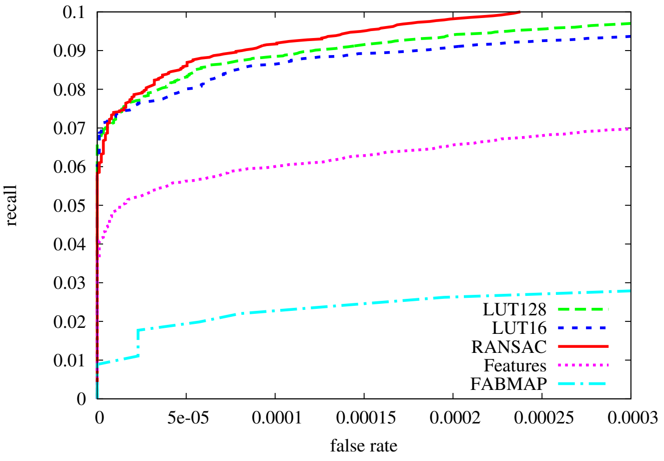
Table 4.2 gives the onsets of the ROC curves. That is: the number of correctly found image pairs, until the first incorrect image pair is found. The FABMAP result seems particularly low, but this is partly artificial because it used only a tenth of the image set.

4.4.6 Results: Precision-Recall curves given pose estimation success

In Figure 4.6, the ROC curves for the RANSAC and LUT methods are plotted based on evaluating the estimated poses. As can be seen, the RANSAC method has a slightly lower false alarm rate for very low recall values. For higher recall values, however, the LUT128 and LUT16 methods outperform the RANSAC method. At 0.07 recall, which amounts to about 32.000 correctly estimated camera poses, the false alarm rate for RANSAC was about 0.00086, corresponding to 750 incorrect poses, while the LUT16 method had a false alarm rate of 0.00033, corresponding to 287 incorrect poses. The LUT128 is slightly better than the LUT16 method for relatively low recall rates up to 0.08.



(a)



(b)

Figure 4.5: The ROC curves given the home recognition application. (a) shows the complete ROC curves. (b) zooms in on the first 10% of the recall with the first .03 % of the false rate.

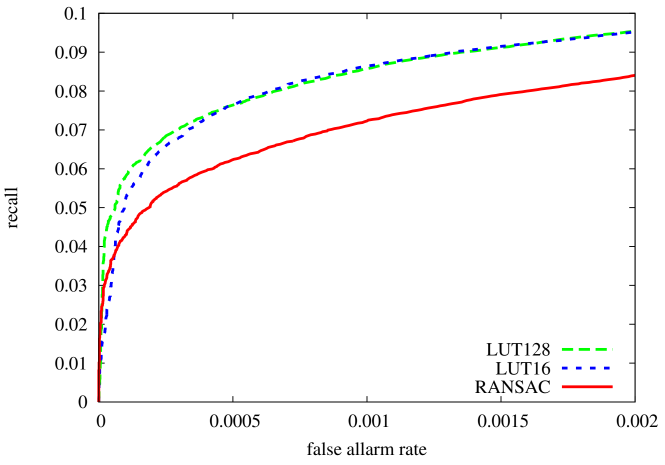


Figure 4.6: The ROC curves based on successfully estimated relative poses.

Table 4.3: Average computational time usage per image pair in milliseconds for the different methods.

Features	RANSAC	LUT128	LUT16
-	0.1	0.2	0.01

4.4.7 Computational time

For all of the similarity measures, the amount of computational time is dominated by the matching of image features which took on average .3 ms per image pair. As explained in Chapter 2 matching techniques - such as the Bag of Words method - can greatly improve the efficiency of matching image features. In Table 4.3, we therefore report only the computational cost of the similarity measure given a set of point correspondences.

Thus, the Feature method, only has to evaluate Equation 4.1 with known values and thus incurs close to zero computational cost. The RANSAC method is faster than the LUT128 method, which seems to contradict the results found in Table 3.1 of Chapter 3. However, in those experiments only geometrically close image pairs were used which on average have much more feature matches. The run-time complexity of the RANSAC method is almost linear in the number of feature matches, while the LUT-based methods have the extra overhead of processing the lookup table. For the LUT16 method, this overhead is clearly much smaller.

4.5 Discussion

The results show that both the RANSAC and the LUT-based similarity measures outperform the FABMAP and the simple feature-based method. The performance of the LUT-based method itself is slightly better than the RANSAC-based method, especially when taking into account its low computational cost.

Nevertheless, the LUT-based similarity measure does not have the advantage of using the total number of features found in the images. The RANSAC and the simple feature-based methods do use this value as a normalization term. We do not apply this same ad hoc normalization technique in the LUT-based similarity measure, in order to keep the similarity values probabilistically sound. Note that extracting two times the number of matches from the total number of found features, results in the number of features found in the images that do *not* result in a match. It would be interesting to investigate how this information of the number of unmatched features can be incorporated in the similarity measure in a probabilistic way.

Indeed, an advantage of the LUT-based similarity measure is that the similarity values are probabilistically relevant. Each similarity value approximates the probability that the ground truth pose falls in the two dimensional histogram bin associated with the estimated pose. The approximation is thus performed by choosing the bin with the highest probability. We could also have chosen to make the measure independent of the bin size of the LUT by, somehow, marginalizing over all bins in the histogram.

4.6 Conclusion

In this chapter, we proposed a method to measure image similarity for view-based mapping. The method is based on the planar pose estimation approach introduced in Chapter 3, which efficiently uses lookup tables (LUT) to compute a discretized density function over the space of possible relative camera poses.

Experiments on the datasets taken in real home environments demonstrate that the performance of the new method is better than the commonly used RANSAC-based similarity measure. The ROC curves show that at zero false rate, the LUT-based method with 128^3 bins finds 14% more links than the RANSAC-based approach. Even with a much smaller LUT with 16^3 bins the performance is still 3.5% better. The experiments also confirm that, for the view-based mapping task, image similarity measures that use geometric constraints, including the RANSAC and the LUT-based methods, outperform feature-based methods, including FABMAP.

However, the main advantage of the proposed method over existing methods is its efficiency. If a small 16^3 bin LUT is used, then the average time to determine a accurate similarity value based on a set of point correspondences is only 10 microseconds. This is 10 times faster than the popular RANSAC-based method. In practice this means that during mapping more images can be compared with newly taken images. This makes it possible to create larger view-based maps.

There is also a theoretical advantage of the proposed method over existing methods. The image similarity measure is based on the ability to estimate the relative pose between the camera poses given two images. Most specifically, the similarity value is the probability that the estimated relative camera pose is correct. This is clearly relevant for applications such as view-based SLAM and view-based robot navigation. Because of this probabilistic nature, it should be relatively straightforward to incorporate it in existing probabilistic frameworks used for geometric mapping.

5 Data association using connected dominating sets

In this chapter, a new and practical method is developed to solve the localization and mapping task for view-based topological maps.[1] Both tasks require an efficient solution for the data association problem as described in Chapter 2. We approach this problem by comparing a new image with only a small subset of the complete set of images in the map. We propose the use of the Connected Dominating Set technique as a solution for choosing such a minimal set of key images that still represents the complete map. This method is used in Appendix B and Appendix C to efficiently perform goal-directed navigation and mapping. It can be combined with the image similarity measure, based on the geometric estimation method described in Chapter 4.

5.1 Introduction

In Chapter 1, we introduced the view-based topological map consisting of a graph. Each node in this graph denotes an image taken by the robot. Each link between two images denotes that they partly depict the same structures of the environment and a relative camera pose up to scale can be estimated. In Chapter 2, we have discussed some image similarity measures that determine whether two images can indeed be linked. In Chapter 4, we proposed our own efficient and robust similarity measure.

The constructed view-based topological map can be used for a variety of tasks, such as goal-directed navigation and human robot interaction, and can be extended using newly acquired images. At the basis of these tasks is the ability to localize where the robot is in the map given a new image. This involves solving the data association problem (Thrun, 2002; Durrant-Whyte and Bailey, 2006). We have seen in Chapter 2 that the data association problem for view-based topological maps involves finding the images in the map that depict an overlapping part of the environment as seen in the new image. In practice, this means that the image similarity values between pairs of images exceed a certain threshold. In this chapter, we will call the images for which this is indeed the case "matching images" or "matches".

View-based topological localization can be defined as finding at least one of the matches given a new image. This match gives a rough idea of the localization of the robot in the map. We will call this type of localization "coarse localization". In most applications, however, we would want to find as many matching images in the map as possible. This is, for example, the case when the map is extended with a new image and we want to

[1]The work described in this chapter was published in the Journal Robotics and Autonomous Systems (Booij et al., 2009).

add links to all matching images in the map. We define this type of localization as "fine localization".

The effort necessary to perform both types of tasks scales linearly with the size of the map, making perfect localization practically impossible for realistic scenarios. Localization for view-based maps can be performed more efficiently by considering only a small selection of previously acquired images that gives a good representation of the complete set. The images in this small selection are called "key" images. For a fine localization this can be followed with an extra step by using the matching key images to search more locally for more image matches.

However, it is unclear what constitutes a good representative subset of a collection of images. In Section 2.3, we have already seen approaches to picking such a subset. For example, by uniformly sampling the images based on elapsed time or robot displacement. A better approach is to use the structure of the graph, effectively performing a kind of sampling in the space of images (Zivkovic et al., 2005; Li and Kosecká, 2006; Valgren et al., 2007). This means that parts of the environment where images are harder to match, for example, because of bad lighting conditions, should be represented with more images than parts where a lot of images match each other.

In this chapter we propose a view-based localization approach based on a graph theoretic method called the "Connected Dominating Set" (CDS) (Guha and Khuller, 1998). A CDS denotes a subgraph that contains the minimal number of nodes that still covers the complete graph. In other words, each node in the complete graph is either *in* the subgraph or linked to one of the nodes that is in the subgraph. This concept is commonly used for broadcasting tasks in large networks of computers. It was first proposed for the task of robot localization in Booij et al. (2006). Later it was used in Anati and Daniilidis (2009) in a robot mapping application and in Snavely et al. (2008) for estimating the 3D geometry of famous buildings or tourist sites.

In Section 5.2, we describe the problem of finding key images given a view-based topological map using a real world example. We then give the definition of the Connected Dominating Set in Section 5.3, and show that it exactly solves the problem of finding key images. In Section 5.4, we describe our algorithm including implementation details to compute an approximate CDS which has a linear complexity with respect to the number of images. In Section 5.5, the CDS algorithm forms the basis for both a coarse and a fine localization method by combining it with an hierarchical data association scheme. For a view-based topological mapping system the CDS algorithm is used as a starting point, by determining a new set of key images for each newly acquired image.

In Sections 5.6 and 5.7, the real home datasets are used to evaluate both the localization method and the mapping method. In both cases, evaluation will focus on the efficiency and accuracy with respect to an exhaustive localization or mapping scheme in which new images are compared with all images. Finally, we investigate how the CDS-based methods compare to other techniques of picking key images. Our conclusions are given in Section 5.8.

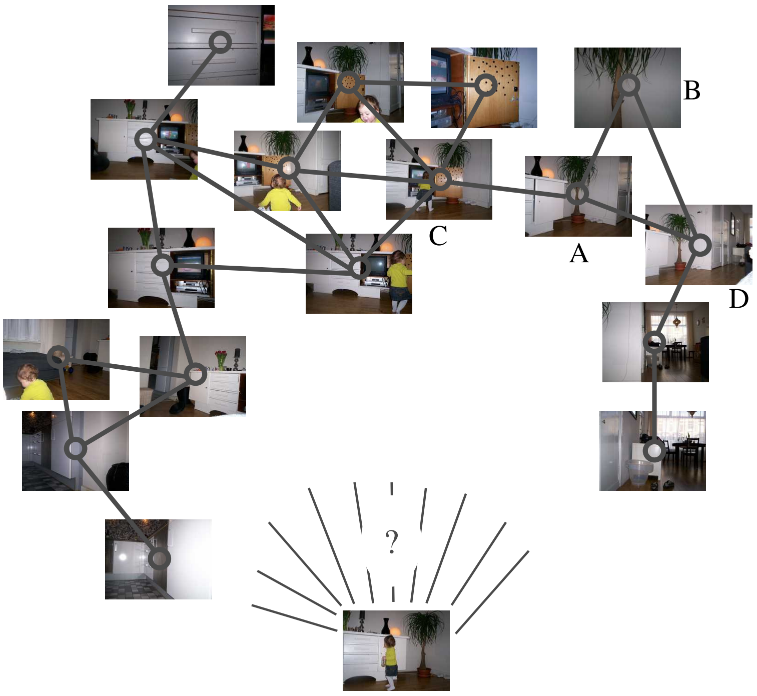


Figure 5.1: An example of a view-based topological map with 17 images taken in a home environment. Images depicting overlapping parts of the environment are linked. A new image shown at the bottom is taken in this same environment. To perform localization based on this new images or add it to the map, matching images have to be found. The problem is: with which of the images in the map must one compare the new image, such that the minimum number of images are compared while finding the maximum number of matches?

5.2 Problem statement

Figure 5.1 shows an example of a view-based topological map and a new image taken in the same environment. To perform localization using this new image, we could compare it with all images in the map. For such a small environment with only a few images this is possible. However, commonly view-based maps contain thousands of images, making it necessary to consider a subset of all images. The problem is with which of the images in the map to compare the new image, such that the minimum number of images are compared while finding the maximum number of matches.

In this example, some of the images have quite some overlap. If two images are indeed very similar, it would suffice to compare the new image with only one of them. This overlap is actually encoded in the links connecting the images. Groups of images that are fully connected to each other will have a large overlap with each other, and could be represented with only a few images. At the same time, images with only few links are rather unique. It is becoming clear that we can use the structure of the graph to pick key images.

A well-known example that uses this structure is the spectral clustering method. In Zivkovic et al. (2005); Li and Kosecká (2006); Valgren et al. (2007) this is used to cluster the graph in a set of subgraphs, each containing images which are visually similar. An image can be picked per cluster, resulting in a set of key images. However, spectral methods are known to be complex and computationally intensive. On top of this, the question remains which images should be chosen to represent each cluster. In the next section we propose a different method that directly gives a set of key images.

5.3 Connected Dominating Sets

To find a set of key images using the structure of the view-based map, we make the assumption that the new image is approximately the same as at least one of the images in the view-based map. This is true if the images taken from the environment densely sample all possible images that the robot could take. If this is the case, then we can define a minimal set of images for which at least one image matches the new image.

If the new image is approximately the same as an image in the map then it will also match all the images that are linked to this same image. If the new image in Figure 5.1, for example is approximately the same as images "A", then it will also match image "B" "C" and "D". Thus, to find a first match for the new image, it suffices to compare it with one of the images "A", "B", "C" or "D" and ignoring the other three. This holds for all images in the map. It would suffice to compare the new image with a set of key images, which has the property that every image in the map is either linked to a key image or is a key image itself.

This is exactly the definition of a Connected Dominating Set (CDS), a concept originating from graph theory commonly used for broadcasting in large networks (Guha and Khuller, 1998).

Remember that the topological map can be seen as a graph $G = (V, S)$, in which a node $v \in V$ represents an image and a link $(u, v) \in S$ represents that the two images which correspond to node u and v match. Two linked nodes are also called neighbors of each other. A Connected Dominating Set V' is defined as follows: the set of nodes in the Dominating Set V' is a proper subset of the original set V, such that every node u in the original set V is either in the Dominating Set V' or linked to a node in V':

$$\forall u \in V : u \in V' \lor \exists v \in V' : (u, v) \in S \tag{5.1}$$

A Connected Dominating Set V' is a dominating set that is also a connected subgraph of the complete graph G, when combined with the existing links S. This also allows efficient path planning and robot navigation to be performed, using the nodes in the CDS.

The problem now is to find a CDS with the minimal number of nodes to compare as few images as possible. Such a CDS is called a minimal CDS. Note that a graph could have multiple minimal CDSs, for which we have to find one. This task is, however, known to be NP-complete. Fortunately, algorithms exist that can find a good approximation and have a computational complexity in the order of the number of nodes of the graph (Guha and Khuller, 1998). Most of these algorithms first remove links to make a spanning tree with as many leaves as possible and then define the set of all non-leaves as the approximate minimal CDS. In the next section, we describe the algorithm used in the experiments.

5.4 Determining a CDS

In Guha and Khuller (1998) a number of algorithms are proposed that find a CDS with close to the minimum number of nodes using computation time in the order of the number of nodes in the graph. We first describe how this algorithm works conceptually and then give a detailed explanation of its implementation.

5.4.1 Approximation algorithm

We focus on one of them, which we name the “GuhaCDS” algorithm[2]. We slightly modify the original algorithm so that it can cope with disconnected graphs. This modification is needed, because in rare occasions the graph is not connected. Usually, this is caused by a single image that did not match any other image, for example, because the view of the camera was blocked by persons walking near the robot.

We explain the GuhaCDS algorithm, using a small example graph shown in Figure 5.2 and a simple coloring scheme:

1. color every node of the graph white (Figure 5.2(a));
2. choose a white node with the highest number of neighbors;
3. color this node black and color all white neighboring nodes gray (Figure 5.2(b));
4. choose a gray node that has the most links leading to white nodes (node 2 in Figure 5.2(c));

[2]This algorithm is called “Algorithm 1” in Guha and Khuller (1998).

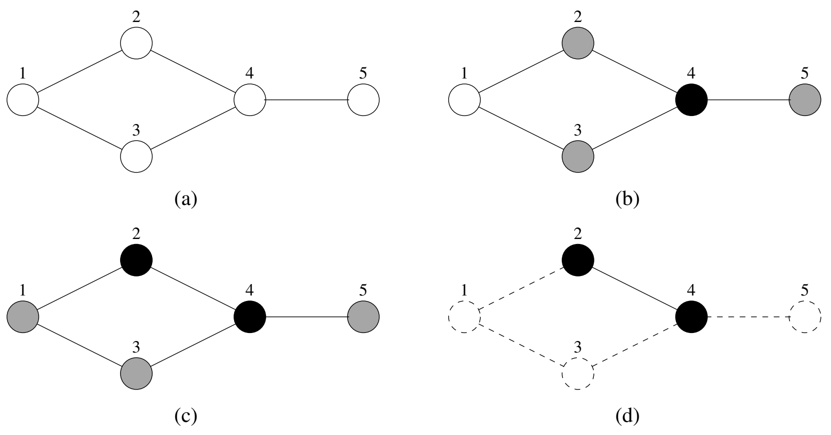


Figure 5.2: A simple example describing the approximation algorithm.

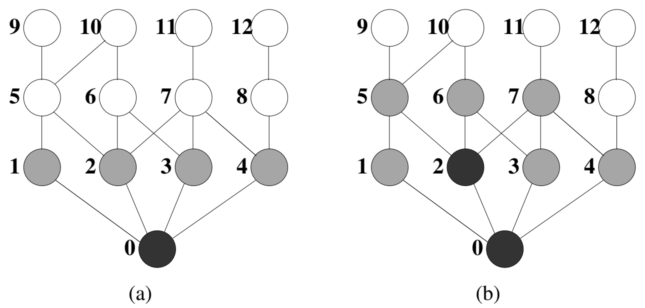


Figure 5.3: A more involved example graph describing the CDS implementation. See the text for an explanation.

5. if no such gray node exists, go to 2;
6. go to 3 until there is no white node left;
7. the black nodes now compose the Connected Dominating Set (Figure 5.2(d)).

Naively implementing this algorithm will result in a computation time that is quadratic or even cubic in the number of nodes. However, it allows for an implementation with linear complexity as explained in the next section.

5.4.2 Efficient implementation

In Guha and Khuller (1998) one can find an explanation of how the GuhaCDS algorithm can be implemented to use an amount of computation time linear in the number of nodes. This is done by using a datastructure that maintains the number of white neighbors for

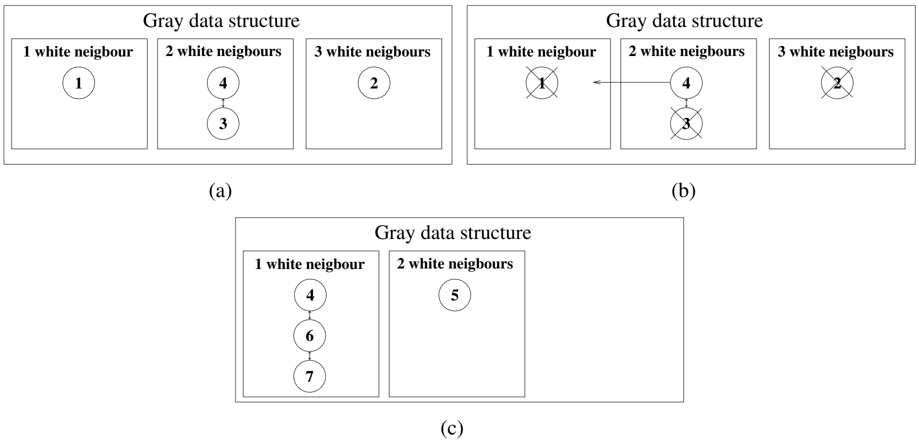


Figure 5.4: The evolving datastructure for the example describing the CDS implementation. See the text for an explanation.

each gray node. The datastructure is implemented as an array of doubly linked lists, where each list contains gray node indices with the same number of white nodes.

As with the algorithm itself, we explain this technique using an example graph shown in Figure 5.3. The changes in the datastructure are shown in Figure 5.4. We use a graph of 13 nodes with some links. At a certain point, node 0 is colored black (Figure 5.3(a)) and thus neighboring nodes {1,2,3,4} are colored gray. The datastructure then contains an array of 3 doubly linked lists, one for gray nodes with 1 white neighbor, one for 2 white neighbors and one for 3 white neighbors (see Figure 5.4(a)). In the next time step, the gray node with the most white neighbors is picked. This amounts to picking one of the nodes in the last list of the data structure. In this example, this is node 2. Node 2 is colored black and all its white neighbors are colored gray (Figure 5.3(b)). The datastructure is then updated accordingly. First node 2 is removed, because it is no longer gray, and nodes 1 and 3 are removed from the datastructure, because all their white neighbors are now gray (Figure 5.4(b)). Node 4 is moved to a different list because it has lost one white neighbor. Then the new gray nodes that have white neighbors are added to the datastructure (Figure 5.4(c)).

Because the datastructure is implemented as an array of doubly linked lists, all adding and removing operations can be performed in constant time. Actually, being perhaps overly specific, the operations are performed in amortized constant time, because the growing array occasionally needs to be relocated in memory. The number of changes needed is a function of the number of gray nodes and the average number of links between the nodes. Thus, the computation time for one time step does not depend on the total number of nodes in the graph. To search for neighboring gray nodes in the datastructure, we maintain an extra array. This array keeps the number of white neighbors for every gray node, so we know in which of the lists to search. To speed this up even further, an array of pointers could be added for each node that points to the specific node in the datastructure.

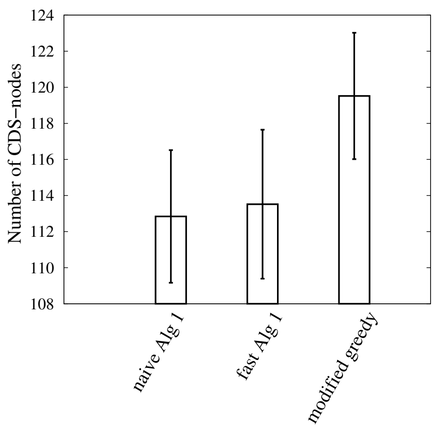


Figure 5.5: Comparison of the size of the CDS for different algorithms. The histogram indicates the mean value and the error bars indicate the standard deviation for 100 random graphs.

5.4.3 Discussion

A variant of the GuhaCDS algorithm is proposed in Guha and Khuller (1998) which we call the "Greedy GuhaCDS" algorithm[3]. Although this algorithm is slightly more involved, it has the advantage that an upper bound is given for the number of nodes of the CDS it computes.

The Greedy GuhaCDS is based on the approach as the GuhaCDS algorithm, but instead of coloring one node black per iteration, it sometimes colors two nodes black, looking one node ahead. This is accomplished by replacing steps 3 and 4 by the following steps:

3a. color the chosen node or the two chosen nodes black and color all white neighboring nodes gray;

4a. choose a gray node or a gray node and neighboring white node, which ever results in the most new gray nodes.

However, the existence of an upper bound for the number of nodes of the CDS does not guarantee that the number of nodes is lower than the simpler GuhaCDS method. In practice this indeed seems not the case. We applied the algorithms including the efficient implementation of GuhaCDS to a set of randomly created graphs. The graphs consisted of a 400 nodes and 1000 links. In Figure 5.5, the average size and standard deviation is shown for the different algorithms. The naive and efficient implementations of GuhaCDS are not significantly different. The Greedy GuhaCDS, however, resulted in larger CDSs. To see the increase in efficiency of the efficient implementation of the GuhaCDS algorithm, see Figure 5.6.

[3]This algorithm is called the "Modified Greedy Algorithm" in Guha and Khuller (1998).

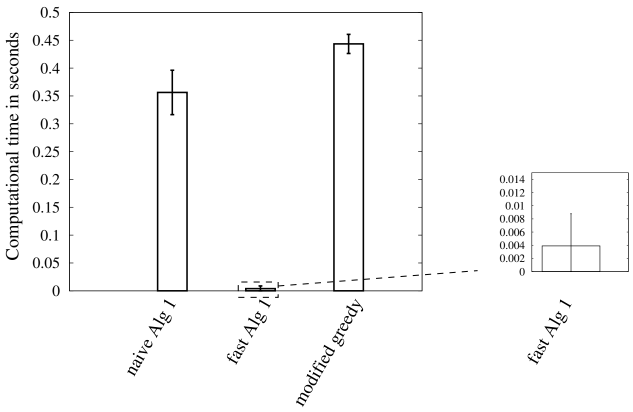


Figure 5.6: Comparison of the computation time used by the different algorithms. The histogram indicates the mean value and the error bars indicate the standard deviation for 100 random graphs.

5.5 Solving localization and mapping

From an existing view-based map, we can now compute a set of key images. This set can directly be used for a coarse localization, by comparing a newly captured image with the images in the CDS and using the matching CDS nodes as the topological localization. This scheme can also form the basis for an incremental mapping scheme which adds images to a growing map. New links are added to the map from new images to matching images that are already in the map. In this section, we propose methods based on the CDS algorithm for a fine-grained localization method and an incremental mapping system.

5.5.1 Hierarchical fine localization

Localization in a view-based topological map was defined as finding those nodes in the graph for which the image matches newly acquired images. By matching a new image with the set of CDS images, we can already find some of theses matches, which would result in a coarse localization. However, sometimes a more fine-grained localization is required, involving more matching images. The matching CDS images indicate where to look for those additional matching images

To determine as many matching image pairs as possible, the new image is compared with all the images that are linked to matching CDS images. Going back to the example given in Figure 5.1: If image A was in the CDS and matched the new image, then images B, C and D are also compared to the new image. See Algorithm 3 for an overview of the complete localization method.

Algorithm 3 Hierarchical fine localization

```
localization $L = (\{\})$
Take a new image $I_c$
$V'$ = computeCDS(G)
**for all** CDS nodes $v'$ in $V'$ **do**
  **if** match$(I_{v'}, I_c)$ **then**
    Add match: $L \leftarrow \{L, v'\}$
  **end if**
**end for**
**for all** nodes $v$ in $V$ **do**
  **if** node $v$ links to a matching CDS node $v' \in L$: $(v, v') \in S$ **then**
    **if** match$(I_v, I_c)$ **then**
      Add match: $L \leftarrow \{L, v\}$
    **end if**
  **end if**
**end for**
```

5.5.2 Incremental mapping

The CDS algorithm can also be used for an incremental topological mapping method. For each newly acquired image, a node is added to the map. The fine-grained localization method is then performed for the new image and links are added from the new node to the found nodes.

This scheme can be used to incrementally map a complete environment repeating the process for each new image. It can, however, be improved by assuming that images are acquired in a sequence. This is often the case in robot mapping scenarios, as explained in Section 2.3. Each new image is usually similar to the previously acquired image. Thus, there is a relatively high chance that it also matches images that matched this previous image. Therefore, we modify the localization step not only to compare it with images that are linked to matching CDS images, but also those linked to CDS images that matched the previously taken image. Pilot experiments have shown that this results in an increase of (on average) 9% of image comparisons. In Algorithm 4, an overview of the complete incremental mapping method is given that uses this modification of the localization step.

Note that the mapping method determines a new CDS for each newly acquired image that best represents the set of images at that moment. This is needed because the minimal CDS can change considerably after adding a new node. It can even lead to a decrease in the number of CDS nodes, because newly added images can potentially represent a much larger set of images than the images taken previously. The incremental mapping methods could be improved by modifying the GuhaCDS algorithm to take advantage of the previously computed CDS. This is not considered in this thesis, since the computation time of the efficient CDS implementation was negligible.

Algorithm 4 Incremental hierarchical mapping

```
graph $G = (V, S) = (\{\}, \{\})$
repeat
  Take a new image $I_c$
  Add current node $c$ to graph $V \leftarrow \{V, c\}$
  $V'$ = computeCDS(G)
  for all CDS nodes $v'$ in $V'$ do
    if match$(I_{v'}, I_c)$ then
      Add link: $S \leftarrow \{S, (v', c)\}$
    end if
  end for
  for all nodes $v$ in $V$ do
    if there is a node $v' \in V'$ that links to $v$: $(v, v') \in S$
          and links to $c$ or $p$: $(v', c) \in S \vee (v', p) \in S$ then
      if match$(I_v, I_c)$ then
        Add link: $S \leftarrow \{S, (v, c)\}$
      end if
    end if
  end for
  Current node becomes previous node: $p \leftarrow c$
until end of mapping
```

5.6 Experiment: view-based localization using real data

The developed localization method based on the CDS algorithm can be used to perform a robot localization on a topological map given a new image. To evaluate the method, we use the image datasets described in Appendix A. The developed method is compared to other view-based localization schemes for both the coarse localization task and the fine-grained localization task.

5.6.1 Aim of the experiment

The CDS method finds images in a large view-based topological map that match with a new image, without comparing the new image with every image in that map. Comparing an image with all images is called “full” localization. One of the main goals of this experiment is thus determining the speed up of the CDS method as compared to full localization.

Because the CDS method does not compare all images with the new image it could be that some actually matching images are not found. This results in a failure to localize the robot. Thus, a second goal is determining the accuracy loss of localization when using the more efficient method. This accuracy depends on the distribution of the CDS nodes in the graph. A good representative set of images would consist of relatively more images from parts of the map that are more difficult to match.

As we have seen in Section 5.1 and Chapter 2, other methods exist to pick a set of key images. We also compare the CDS method with some of these approaches.

5.6.2 Evaluation measures

For evaluation purposes, we need to measure the speed-up of the proposed localization method with respect to the exhaustive localization scheme, as well as the accuracy of localization.

The computation time spent by any view-based topological localization scheme is highly dependent on the used image comparison measure. However, in this chapter we treat the algorithm to compute this comparison measure as a black box and assume that every image comparison costs a fixed amount of time. The speed-up of an efficient method with respect to comparing all images in the map is defined by the ratio of number of performed images comparisons:

$$\text{speed-up}_{\text{loc}} = \left(\frac{n_{\text{map}}}{\#\text{comparisons}} - 1 \right) \times 100\%, \tag{5.2}$$

where n_{map} is the number of images in the map.

For the coarse localization task, it is necessary to find at least one image in the map that matches a new image. This can either fail or succeed. The percentage of success for a number of coarse localization runs is defined as the “coarse localization accuracy”:

$$\text{coarse localization accuracy} = \frac{\#\text{tests in which a match was found}}{\#\text{tests}} \times 100\% \tag{5.3}$$

For fine-grained localization, the accuracy is measured by the average number of found image matches as compared to the total number of matches found by the full localization:

$$\text{fine localization accuracy} = \frac{\#\text{found matches}}{\#\text{matches from full localization}} \times 100\% \tag{5.4}$$

5.6.3 Data

Experiments are carried out on the challenging data sets acquired in the real home environments described in Appendix A. To evaluate localization, we need to obtain topological view-based maps and new images taken in the same environment. This is achieved using a leave-one-out approach. From an image set, one image is taken for localization and the remaining images are used to build a topological map. Localization on this map is then performed using the left out image. This scheme is then repeated for all the images in the data set.

As we know there are various approaches to building a topological view-based map. In this experiment, we simply compare all the images of a set with each other and create a link for matching images. In the next section, we will explore the more efficient map building method based on the CDS.

For the map building process and the localization method, we need to define an image comparison measure. The CDS data association scheme can be combined with any image comparison technique. In the experiments in this chapter, we use a straightforward method based on corresponding local image features and imposing the epipolar constraint as described in Section 2.2. We now repeat this description briefly.

First, a set of point correspondences is found similar to experiments described in Section 3.7. The omnidirectional images are mapped to panoramic images (Bunschoten, 2003), from which feature points are found using the Scale Invariant Feature Transform (SIFT) (Lowe, 2004). Features are described by the standard SIFT descriptor of 128 dimensions. Point correspondences between two images are determined by applying the standard matching scheme as described in (Lowe, 2004).

Second, the relative robot pose is computed from these correspondences, using a state of art method, explained in Section 3.5. Two images match if the number of point correspondences that fit this pose, divided by the lowest number of features found in the two images, is larger than a certain threshold. Pilot studies in an office environment with a threshold of 0.1 resulted in a lot of good matches and no false matches.

The resulting topological maps are visualized in Figure 5.7 as connectivity graphs, linking the matching images and using hand corrected odometry and GPS information for the position of the image-nodes. Note, however, that the odometry information is not used by the proposed method. As can be seen, for all datasets, a lot of images matched. Figure 5.8 visualizes the resulting graphs of the Almere 4 set as a connectivity matrix, which shows more clearly the loop closing image matches by the off-diagonal non-zero values.

We highlight some of the characteristic parts of the home environments as depicted in Figure A.2 and discussed in Appendix A. In the connectivity graphs (Figure 5.7), the robot positions of the example images are visualized with a “D”, for images that are difficult to match, and an “E”, for images that are easy to match.

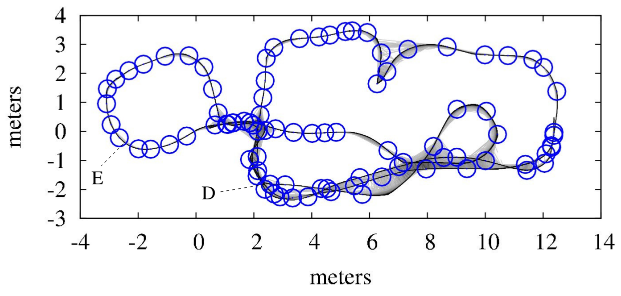
5.6.4 Results: distribution of CDS nodes

The CDS algorithm is applied to the three datasets using the leave-one-out method. The resources used by the algorithm are negligibly small compared to the time needed for actually comparing images. For all the datasets, the computation time was always smaller than 10 ms.

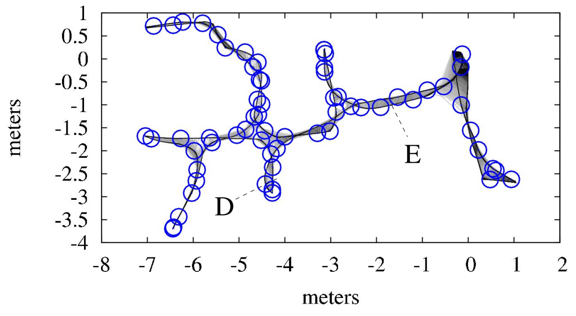
In Table 5.1 the number of CDS nodes is given averaged over the different runs. For typical runs, we show in Figure 5.7 the distribution of the key nodes in the maps. One can see that relatively more key images were picked in the neighborhood of the difficult images close to images labeled with a “D”, than parts of the environment where good images were acquired close to images labeled with an “E”.

5.6.5 Results: comparison with exhaustive localization

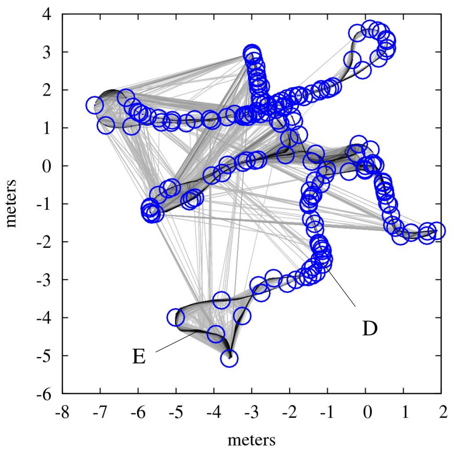
In Table 5.1, the average speed-up and accuracy for both the coarse and fine localization are shown. For all datasets, the accuracy is not 100%. This means that some of the images from the dataset did not match any of the CDS nodes based on the remaining images. For the Spaan 1 this was the case for 12 images. In Figure 5.9, the position of these images in the map is shown. In general, they were taken at the geometrical border



(a) Almere 4

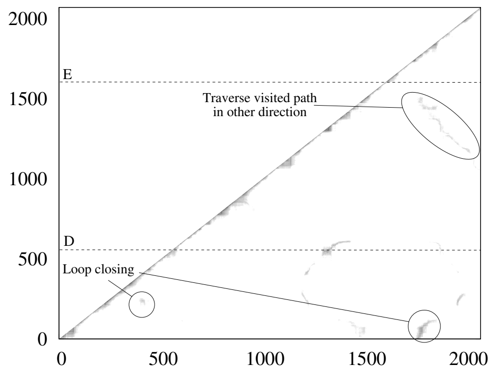


(b) Spaan 1



(c) Biron 1

Figure 5.7: The topological view-based maps visualized as connectivity graphs. The positions of the nodes were determined using the ground truth data. Each printed line represents a link between the nodes. The nodes themselves are actually not shown to prevent the figures from getting cluttered. Circles denote the nodes that are in the final CDS. The nodes indicated by a “D” correspond to example images that are difficult to match, plotted in the left column of Figure A.2. Nodes indicated by an “E” are easy to match and plotted in the right column.



(a) Almere 4

Figure 5.8: The topological view-based maps visualized as connectivity matrix of the Almere 4 set. Image pairs with a higher similarity values are represented with darker pixels. The entries on the main diagonals are the result of matching sequential images, while the off-diagonal entries reflect instances of loop-closing.The "D" and "E" again indicate the difficult and easy images shown in Figure A.2.

Table 5.1: Speed-up and accuracy of the CDS-based localization methods for the real home datasets.

		Almere 4	Spaan 1	Biron 1
# images		2071	1436	1674
# cds-nodes		88.0	65.0	161.0
coarse localization	% speed-up	2251.7	2109.1	939.1
	% accuracy	99.7	99.2	95.0
fine localization	% speed-up	1059.1	940.4	637.4
	% accuracy	95.2	92.6	81.2

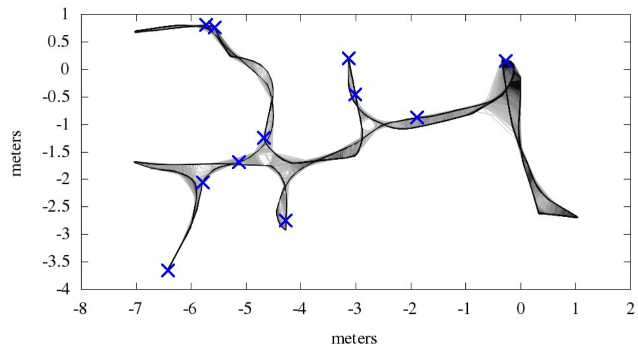


Figure 5.9: The location of images that could not be used for localization in the topological map of the Spaan 1 data set.

of the map or at places where the robot took a turn resulting in severe motion blur (see Figure 5.10).

5.6.6 Results: comparison with other sampling methods

We compared the proposed method with other methods to pick key images. This was done only for the Spaan 1 dataset. To make the comparison as fair as possible, we set the sampling density for each method such that the average number of key images was about 65. This is similar to the number of CDS images (see Table 5.1). Thus the speed-up of coarse localization is more or less equal among the methods. The following methods were used:

Random picks 65 images randomly from the image set.

Position uses the odometry measurements to sample over displacements of the robot.

Time samples images over time.

In Table 5.2, the results are shown for both coarse and fine localization using the different methods for the Spaan 1 set.

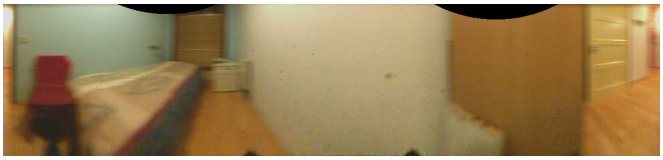


Figure 5.10: Example image from the Spaan data set that has severe motion blur. This is one of the 12 images that was not localized by the CDS-based localization method.

Table 5.2: Comparison of the speed-up and accuracy of localization combined with different sampling methods applied to the Spaan 1 dataset.

		Random	Position	Time	CDS
coarse localization	% speed-up	2107.7	2041.8	2107.7	2109.1
	% accuracy	83.7	90.8	96.4	99.2
fine localization	% speed-up	1150.3	1011.8	998.1	940.4
	% accuracy	72.5	82.7	87.0	92.6

5.7 Experiment: mapping using real data

In addition, to localization the CDS algorithm was used to define a new view-based mapping method. This is summarized in Algorithm 4. This method is applied to the real home datasets. The resulting maps are compared with both maps resulting from exhaustive comparing images and maps based on other sampling.

5.7.1 Aim

The aim of the proposed mapping method is to reduce the computation time spent on mapping, while still finding approximately the same map as found when exhaustively comparing images. However, we have seen in the localization experiment that fine localization does not find 100% of all the matching images in the map. The proposed mapping method is based on the same CDS algorithm. Thus, for each image that is added to the map, the mapping method will miss some of the matches.

In general, the number of nodes the CDS will increase, while the robot is mapping. As a direct result the number of image comparisons will also increase. If, however, images are added that look similar to images that are already mapped, the size of the CDS should stay more or less constant. We test this with an additional dataset, where the robot traverses the same route through the environment twice.

In this experiment, we evaluate the computational speed-up of the proposed mapping method and compare the resulting map to a map that results from comparing all images (as used in the localization experiment).

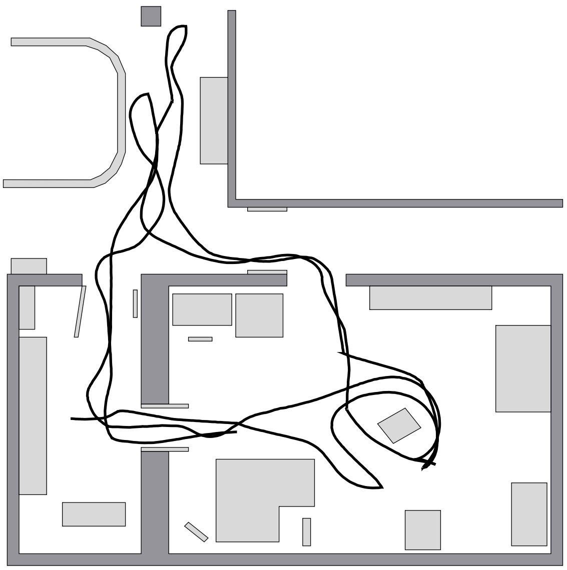


Figure 5.11: Ground floor maps of the Robot Lab environment. The trajectory of the robot is indicated by the black line. The position of the furniture and walls is approximate.

The incremental mapping scheme can also be combined with other sampling techniques which were described in the localization experiment. We compare the resulting maps using these different techniques with the one resulting from the CDS-based approach.

5.7.2 Evaluation measures

Again, the methods are evaluated base on the speed-up and the accuracy as compared to an exhaustive matching scheme. The accuracy is defined as in Section 5.6.2 and the speed-up is defined as:

$$\text{speed-up}_{\text{map}} = \left(\frac{\frac{1}{2} n_{\text{map}} (n_{\text{map}} - 1)}{\#\text{comparisons}} - 1 \right) \times 100\%, \tag{5.5}$$

with n_{map} again the number of images in the map.

5.7.3 Data

The mapping methods are applied to the real home datasets using the same image similarity measure as used in the localization experiment, see Section 5.6.3. In addition, a dataset is used in which the robot traversed the same route twice, which we name the “Robot Lab” set. This dataset was acquired in the university building using the same robot and vision system as used for the Almere 4 and Spaan 1 datasets. See Figure 5.11

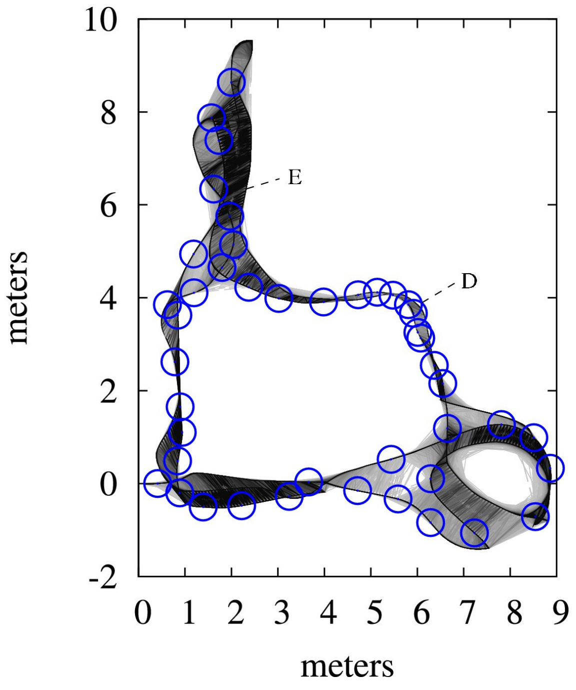


Figure 5.12: The topological view-based map of the Robot Lab, visualized in a similar manner as the real home maps in Figure 5.7.

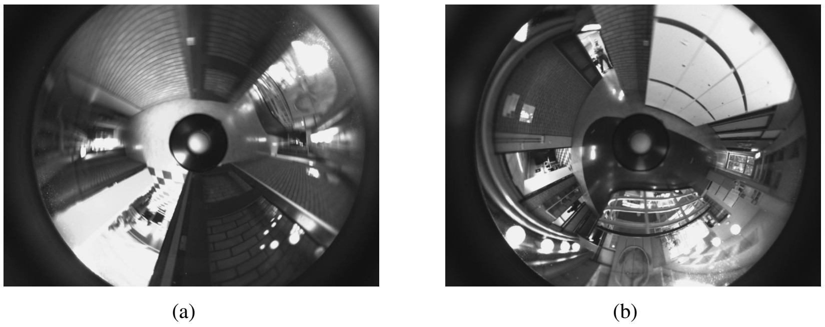


Figure 5.13: Two example images from the Robot Lab set.

Table 5.3: Speed-up and accuracy of the CDS-based mapping method for the real home datasets and the Robot Lab dataset.

		Almere 4	Spaan 1	Biron 1	Robot lab
mapping	% speed-up	997.5	948.2	720.6	759.8
	% accuracy	96.2	96.9	89.7	98.4

for a drawing of the environment and the path that the robot took. As can be seen, the robot drove the same loop in the environment twice. In Figure 5.12, the graph resulting from exhaustive image comparison is shown. In Figure 5.13, two typical images from the data set are layed out.

5.7.4 Results: comparison with exhaustive mapping

The proposed incremental mapping method based on the CDS algorithm is applied to the four datasets. The connectivity graphs and connectivity matrices of the resulting view-based maps are visually indiscernible from the ones computed using the exhaustive method. They are therefore omitted. In Table 5.3, the speed-up and accuracy of the CDS-based method are given. As can be seen, for most datasets the CDS-based method is about 10 times faster than the exhaustive method resulting in only 2 to 4% fewer links. The Biron 1 set, however, proves more difficult. This is most probably due to the false image matches, which misleadingly raise the number of links that exhaustive method finds.

5.7.5 Results: comparison with other sampling methods

We also combined the incremental mapping scheme with the other sampling methods that were used in the localization experiment. They were applied to the Spaan 1 dataset. To make the comparison as fair as possible, we set the sampling density for each method such that the number of images-pairs that is compared is more or equal to the number of image-pairs compared by the CDS method. Thus, the CDS method will use less or equal the amount of computation time. This resulted in somewhat different sampling densities than used in the localization experiment:

Random During each iteration, a new set is chosen with an average number of images equal to .063 times the number of images in the map.

Position After each 43 cm, an image is added to the set of key images.

Time After each 3.8 seconds, a mapped image is added to the set of key images.

In Table 5.4, the accuracy of all methods is shown. Note that experiment was performed such that the speed-up is more or less the equal, and therefore omitted from the table. On top of this, the number of key images that was computed when adding the last image is given.

Table 5.4: Comparison of the accuracy of mapping combined with different sampling methods applied to the Spaan 1 dataset.

method	final # key images	% accuracy
Random	92	71.0
Position	82	82.9
Time	77	85.0
CDS	65	96.9

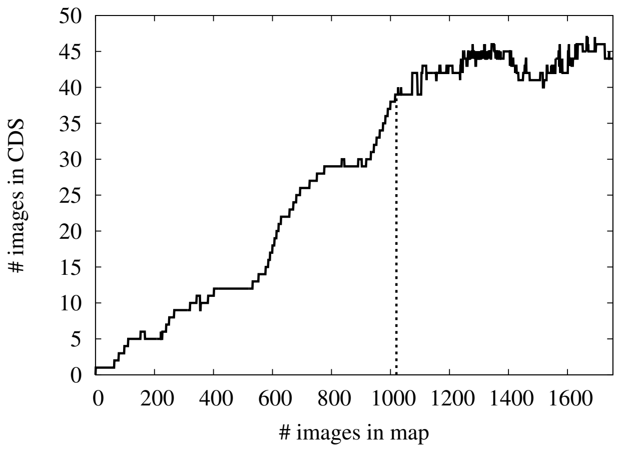


Figure 5.14: The number of nodes in the CDS, while the map of the Robot Lab is growing. The vertical dashed line indicates the beginning of the second traversal of the same loop.

5.7.6 Results: revisiting places

During mapping the environment, more and more images are added to the map and thus the size of the set of key images grows. This is depicted in Figure 5.14. At image 1020, the robot had finished its first loop in the environment with a CDS size of 39 images. During the second traversal of the loop, new images were matched with images taken in the previous loop. This way the robot created links between these images as shown in Figure 5.12. Because of these links only a few extra nodes were added to the CDS during this second loop resulting in a total of 44 nodes in the final CDS (which are indicated in Figure 5.12).

Note that the set of 44 key images of the final map are not composed of the 39 key images of the first loop and 5 extra images of the second loop. The best set of key images is determined for each new image that is added to the map. Images of the second loop might better represent images taken of a particular part of the environment, making

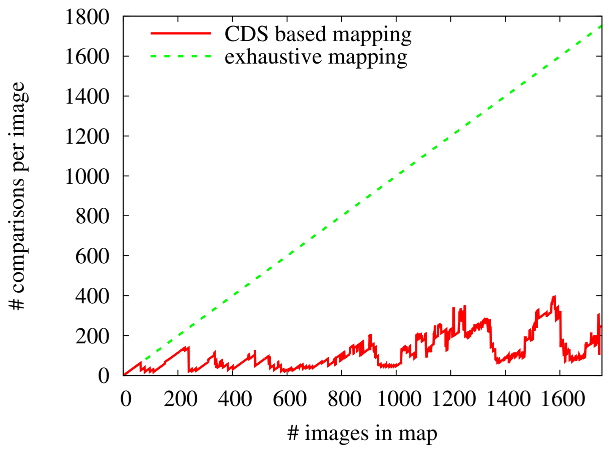


Figure 5.15: The number of performed image comparisons, while building the map. For exhaustive mapping the number of comparisons grows linearly with the size of the map. For the CDS-based mapping, it grows much slower. The fluctuations in the graph of the CDS are mostly caused by the variable speed of the robot. When the robot is moving slowly, then relatively many images will match.

images of the first loop redundant. In the Robot Lab set the final CDS is composed of 11 images of the first loop and 33 of the second loop.

Figure 5.14 also shows that in some occasions the number of nodes in the CDS decreases. This happens when a new image is added which matches already mapped images that did not match each other. This indicates that new images can represent an existing set of images better than the previous key images taken from the set.

Figure 5.15 shows the number of image comparisons performed while the robot is mapping the Robot Lab set. The other datasets resulted in similar plots. As can be seen, the number of comparisons for the exhaustive data association scheme increases linearly with the number of images in the map. The number of comparisons performed by the CDS method barely increases.

5.8 Conclusion

In this chapter we proposed efficient localization and mapping methods for view-based topological maps. Our approach is based on the fact that we consider only a selection of the previously acquired images for matching new images. The selected set of representative images covers the complete set of previously acquired images. We can efficiently detect loops in the trajectory of the robot. We have shown that the problem of finding the minimal number of key images is equivalent to finding the smallest Connected Dominating Set (CDS).

The experimental results show that our method leads to a more efficient distribution of key images. From areas in the environment that are harder to match, for example, because of bad lighting conditions, more images are picked. In this way, loop closure is much more robust, even if it occurs in such an area.

The CDS method is built in a hierarchical data association scheme that incrementally builds a map without using any prior knowledge about the environment. The set of representative images is dynamic. After each newly acquired image, a CDS is determined that best represents the set of images at that moment.

The method is applied to the challenging datasets acquired in home environments. In all datasets, our method finds approximately the same map as is formed in the "full" case that compares all image pairs. However, only 13% of the computation time is used.

Our method outperforms other known sampling techniques, which find larger sets of key images, but fewer matching image pairs, in the same amount of computation time. Our method finds 97% of the matches that were found with a "full" method while position and time-based methods found less than 85%.

Although the CDS method was used stand alone during the experiments, it could just as well be merged with other sampling techniques. For example, it could use the navigation prior of a geometric mapping method as explained in Section 2.4.3. Furthermore, the efficiency of the CDS method could be even further improved by additionally using a more efficient image similarity method like the recently proposed hierarchical methods (Sivic and Zisserman, 2003) or a bag-of-words (Cummins and Newman, 2008) as described in Section 2.2

In the experiments, we used image sets in the order of a few 1000 images. For such dataset sizes, exhaustive data association used for evaluation is still possible, though time consuming. Using the CDS method datasets can scale up by a factor of 10. In Esteban et al. (2008) we used a SLAM system to build a map with more than 10,000 images, implicitly using the CDS method for data association.

If even larger maps need to be constructed, then it would perhaps be beneficial to reduce the number of images in the map. The question then is which images are redundant and can be removed, without changing the map too much. An answer in line with the reasoning in this chapter, is that we should define redundancy as images that look very similar to other images in the map.

6 Conclusion

View-based mapping is a straightforward and relatively easy approach to building a map of an environment. Nevertheless, to build a view-based map of an inhabited home environment in real time, requires both low computational cost and high robustness. In the preceding chapters we have investigated methods that try to meet these requirements. This resulted in new solutions for the problems faced in view-based mapping. We evaluated these by comparing them with existing methods. In this final chapter we summarize our main contributions and experimental results. We are then able to answer the research questions which were formulated in Chapter 1. We conclude this chapter with possible directions for future work.

6.1 Contributions

We began, in Chapter 2, by giving an overview of existing work in the field of view-based mapping. We identified that image similarity measures for view-based mapping are commonly based on the ability to estimate a relative camera pose. Sometimes this is done explicitly by combining photogrammetric techniques with the RANSAC algorithm. However, in a lot of cases it is performed implicitly, and perhaps unintentionally, using less well-founded heuristics. In addition, it is often assumed that the motion of the camera is restricted, because the robot is driving over a planar surface. We showed that this restriction is usually implemented in a rather ad hoc manner.

In Chapter 3, we investigated *how the planar motion assumption can be used to estimate the relative pose given two images*. We proposed two new methods that perform such an estimation.

First, we derived the closed form Planar Two-point algorithm that can compute the relative pose using only two point correspondences. In previous studies on this topic, it was assumed that the relative planar pose can always be determined from two point correspondences. We discovered that this is not true and show that, actually, in 50% of the cases there are two possible poses. In these cases, the Planar Two-point algorithm returns them both. The Planar Two-point algorithm has to be combined with RANSAC to effectively cope with noisy correspondences and mismatches. However, experiments showed that, in practice, the combination of RANSAC with the existing Planar Three-point algorithm is more accurate.

Second, we developed an alternative approach to cope with noisy correspondences and mismatches, based on lookup tables (LUT) and a discretization of the solution space. It was shown that this approach was up to 100 times more efficient while being up to 20% more accurate than the RANSAC-based approaches. The accuracy improvement was most apparent in environments with difficult lighting conditions.

What sets our LUT-based method apart from other methods, is that it directly learns a mapping from local image features seen in the environment to the space of possible planar relative robot poses. This leads to the advantage that it does not require an explicit model of the used camera system, the pose of the camera on the robot, or specific characteristics of the environment, including moving objects and persons. As a result the method is easy to implement and use in different environments. Because the learned model is represented as a 3D lookup table (LUT), the estimation process is reduced to simple lookups, resulting in low computation times.

Furthermore, we investigated *if the proposed methods, based on planar motion, are more accurate and efficient than existing unrestricted pose estimation techniques*. Evaluation on the home environment datasets showed that this was indeed the case. The combination of RANSAC with an unrestricted pose estimation method had a median rotation and heading error of more than .9 radians for each of the datasets. The methods based on planar motion had median error around .4 radians for rotation and around .6 radians for heading. In addition, the used computation time for the proposed methods is 3 to a 100 times less than the unrestricted pose estimation.

In Chapter 4, we investigated *how the uncertainty of the estimated relative pose can be measured*. The LUT-based method outputs a full discretized density estimate of the relative pose instead of just the maximum likelihood solution as estimated by RANSAC-based methods. By applying Bayes' rule it is possible to estimate the uncertainty of the estimated relative pose. We used this uncertainty as a new image similarity measure for the view-based mapping application.

The question was *if this uncertainty measure provides an efficient and robust image similarity measure for the view-based mapping application*. Applying this measure on the home environment datasets resulted in a view-based map with more links than other popular similarity measures. Depending on the size of the LUT, 3.5% to 14% more correct links were found than with a RANSAC-based measure. Moreover, the proposed method is up to 10 times more efficient.

Regardless of the efficiency of the image similarity measure, the computational complexity of view-based mapping increases linearly with the amount of images in the map. In Chapter 5, we investigated *how we could limit the number of image comparisons necessary for view-based mapping*. We proposed a hierarchical data association scheme which exploits the topological structure of the map, limiting the number of necessary image comparisons. The main contribution is the use of the graph theoretic Connected Dominating Set (CDS) to pick a minimal subset of images to represent the complete set of previously taken images. This is used to roughly but efficiently determine the location in the view-based map. For the real home environments this mapping method proved to be more than 7 times faster than exhaustively comparing with the complete set of images.

The last research question was *what effect comparing with only a subset of images has on the resulting map*. For the real home environments, the method finds 97% of the links found by the exhaustive scheme, resulting in approximately the same view-based map.

Overall, the proposed methods achieve a level of robustness and efficiency that allow for a mobile robot to map a real inhabited home environment in real time. Appendix B shows how a robot can use such a map to perform goal directed navigation, while humans walk in close proximity. Here the CDS technique is used for view-based localization. In Appendix C, a more complete robot system is shown which is guided by a human while exploring a home environment. Using the efficient techniques described in this thesis, the robot can build a map of this environment in real time, allowing it to resolve ambiguities by interacting with the guide. These results make it clear that the achieved efficiency and robustness of the proposed methods are not merely quantitative improvements, but essential to make new robot applications possible.

6.2 Discussion and future work

The methods proposed in thesis, could contribute in two different ways to major breakthroughs in the field of robotic mapping.

The first one is rather pragmatic. We have shown in different experiments that our newly developed methods are quantitatively more accurate and more efficient than established view-based approaches. Moreover, most of those approaches have existed for quite some time and have been fine-tuned for various scenarios and applications. For the methods proposed in this thesis, this is not the case. We think that with little effort, the results as reported in this thesis can be further improved upon.

At the end of each chapter we have already mentioned several possible improvements. We think that the largest gain in efficiency can be achieved by more carefully dealing with the memory management of the LUT-based algorithm. It should be implemented in such a way, that the minimal amount of data has to be copied from the memory to the cpu and vice versa.

This gain in efficiency and robustness makes it possible to create bigger view-based maps and map environments under more challenging circumstances than existing systems.

The second opportunity has a more qualitative nature. The developed image similarity measure and the relative planar pose estimation method from which it was derived, both have a strong probabilistic foundation. The ability to efficiently estimate a full density estimate over all possible relative poses is unique. Because of this property it is relatively straightforward to combine it with existing probabilistic methods.

A perhaps obvious example, which was already mentioned in Section 3.8, is the integration of the relative pose estimator with existing geometric mapping systems. These systems often have a strong probabilistic nature and require as input not only estimates of the relative poses, but also estimates of the uncertainties of those poses. These uncertainties are usually modeled as covariance matrices, which are kept fixed for each pose, or are estimated by using first order error propagation techniques. Using our pose estimation method a much more realistic uncertainty can be obtained as input for the geometric mapping system. If needed a covariance can be estimated by fitting a Gaussian on the discretized density estimate in the neighborhood of the maximum likelihood solution. This would improve both the accuracy and the consistency of the geometric map. In combination with the image similarity measure it would lead to the ability to build geometric view-based maps of real home environments.

Another example is the straightforward manner of incorporating prior knowledge of the relative pose into the pose estimation method and the image similarity measure using prior probability distributions. A more challenging task would be to integrate the proposed methods with popular probabilistic bag of words-based image similarity measures, such as FABMAP. This would be a big improvement over the ad hoc geometric check that is now being applied.

The view-based data association scheme, based on the CDS algorithm, did not benefit from the probabilistic nature of the image similarity value. It is a purely graph-based algorithm, and picks key frames only on the basis of image matches, which were obtained by thresholding the similarity values. It would be interesting to try to define a similar data association scheme that uses the similarity values themselves. This would give the ability to make a trade off between the number of performed image comparisons and the probability of missing a certain link between two images.

A Real home datasets

In order to evaluate all the methods described in this thesis we acquired datasets in three different real home environments[1]. Here, we give a brief description of these datasets. In (Zivkovic et al., 2008) the datasets are presented in more detail.

Two different wheeled robots were used to acquire the datasets, namely a relatively small Nomad Scout, and a bigger PeopleBot called "Biron" from the Faculty of Technology of the Bielefeld University (Haasch et al., 2004), see Figure A.1. Both robots were driven through the homes by tele-operation.

To acquire images the robots were equipped with an omnidirectional vision system consisting of a conventional Firewire camera pointing upwards to a convex hyperbolic mirror from Accowle. For both the camera and the convex mirror an accurate model was available, making it possible to relate image pixels to image-rays. On the Nomad the camera was placed at about 1 meter height, while on Biron this was about 2 meters.

In addition, a Sick active laser scanner was mounted on both robots. The laser scans and the odometry of the robot were used to obtain ground truth robot poses by applying a line-based SLAM algorithm described in (Folkesson et al., 2005).

All three homes were relatively small apartments with a single floor. Although multiple datasets were shot in each home, in most of the thesis we use only a single dataset from each of them. In Figure A.3 the trajectories of the robot are shown for these datasets and Figure A.2 shows some typical omnidirectional images from the acquired image sets. In the following we describe the distinct characteristics of each home and the dataset that was acquired.

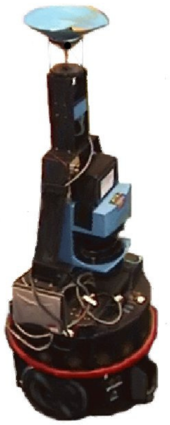
Almere

This house was actually not inhabited but used as a demonstration house by the company "Unet". The dataset we use, named "Almere 4", was captured during day time with the blinds open. This caused some images to be very bright, see Figure A.2(b), while others not being in direct view of a window to be quite dark, see Figure A.2(a). As can be seen in the images the dataset was taken while people walked in close proximity to the robot. For this dataset the Nomad Scout was used which acquired 2071 images taken at 7 Hz.

Spaan

This small student apartment was visited during evening hours and was therefore relatively dark. This resulted in somewhat dark images and, more importantly, motion blur during sharp turns, because of the required higher camera exposure times, see Figures A.2(c)-(d). As can be seen the house itself was relatively feature rich. We use set "Spaan 1" for the experiments in this thesis, which consists of

[1]The datasets, including images, odometry, sonar and laser range data (all time-stamped), are available from `http://www2.science.uva.nl/sites/cogniron/`


(a) Nomad Scout

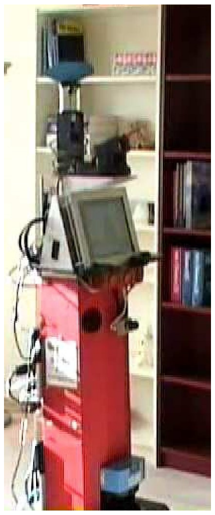
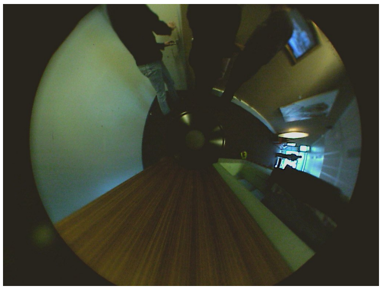

(b) The PeopleBot "Biron"

Figure A.1: The two robots that were used for recording the datasets.

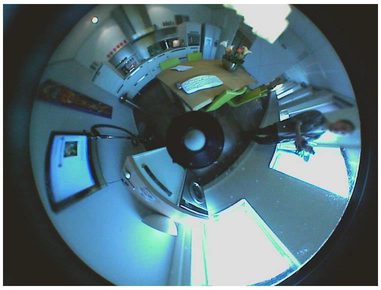
1436 images taken at 5 Hz by the Nomad Scout. In this set there were no people walking in the home.

Biron

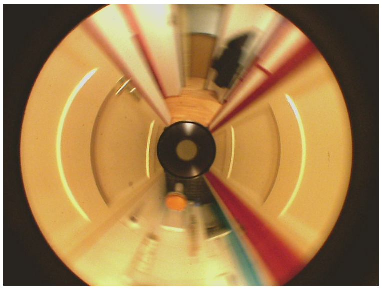
This house was situated in Bielefeld and the datasets were acquired by the Biron robot. The house was only partly inhabited, and some rooms did not contain any furniture. Like the Spaan house, this house was also visited during evening hours. It is the biggest house with 8 different rooms and some narrow corridors, see Figure A.2(e)-(f). While taking this dataset the Biron robot was heavily loaded, making its movement jerky causing in motion blur in the images. As a result the datasets taken in this house were more challenging than the others. We use set "Biron 1" for the experiments which consisted of 1734 images taken at 2.5 Hz and did not contain people walking in the home, except the person tele-operating the robot.



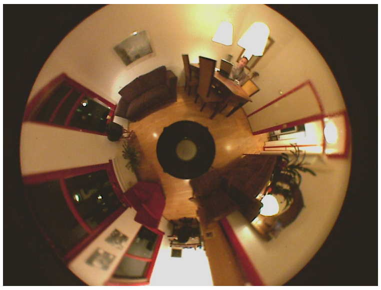
(a) Almere 4, dark



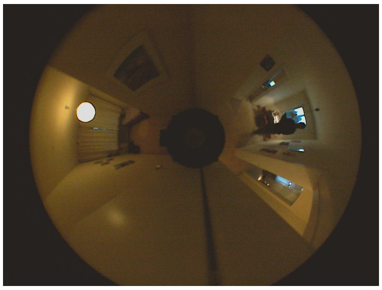
(b) Almere 4, bright



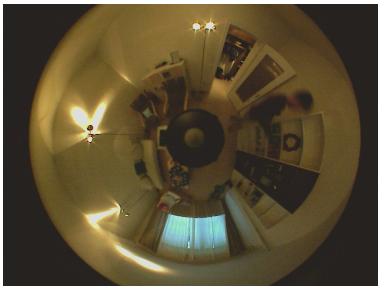
(c) Spaan 1, motion blur



(d) Spaan 1, feature rich

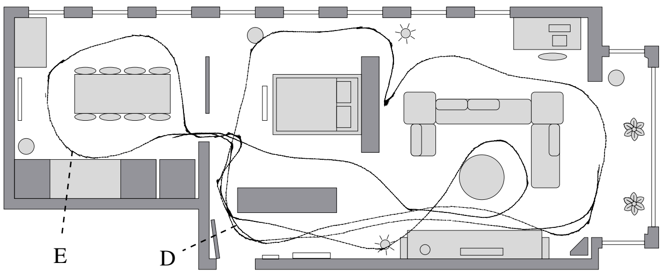


(e) Biron 1, small hallway

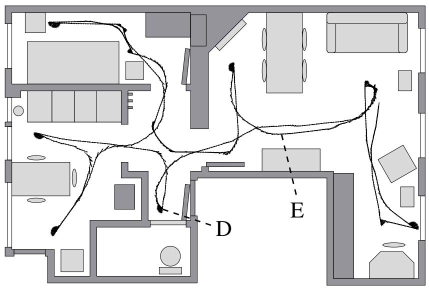


(f) Biron 1, big living room

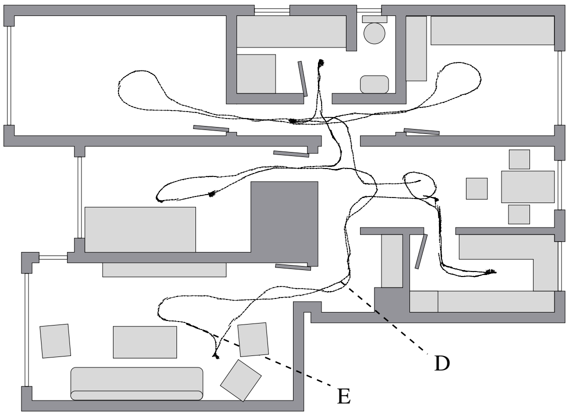
Figure A.2: Example images from each dataset taken by the omnidirectional vision system. From each indoor dataset there are two images. On the left there are images with few distinctive image features and on the right images with many image features. In Figure A.3 the positions of the robot are indicated while taking these particular images.



(a) Almere 4



(b) Spaan 1



(c) Biron 1

Figure A.3: Ground floor maps of the indoor environments with the approximate positions of the furniture. The trajectory of the robot, as estimated by the laser-based SLAM method, is indicated by the black line. The "E" and the "D" indicate the positions from which the feature rich and the feature poor images were taken respectively, shown in Figure A.2.

B Navigation using an appearance-based topological map

In this thesis we have shown how to build a topological view-based map. Although this map does not contain any positional information accept for pairwise relative poses, it can be used for path planning and goal-directed navigation. In this appendix we show that these tasks can be performed robustly and efficiently in a relatively small office environment. The used methods do not involve the LUT-based relative pose estimation or the Connected Dominating Set based localization procedure as described in this thesis. It is however straightforward to incorporate these techniques, which would allow for path planning and goal-directed navigation in more challenging environment, such as homes.

This appendix was presented earlier as a paper at the International Conference on Robotics and Automation in 2007 and was published in its proceedings (Booij et al., 2007). It contains overlap with the rest of the thesis, most notably with Chapter 3. Some inconsistensies with that chapter are explained in footnotes.

B.1 Introduction

Recent developments in the field of 'personal' robots, which interact with humans in a natural way, bring new insights in the representations needed by the robot to fulfill its task.

For example the internal model of the environment, used by the future home robot for goal-directed navigation, must contain spatial concepts understandable for the human. In an indoor environment typical classes such as 'rooms', 'objects' or 'doors' must be detected.

The sensing system of the robot must be able to distinguish between different instances of these classes. The traditional sensors on robots such as laser range finding — mainly used for obtaining geometric information and navigation — have been used for semantic labeling of places (Stachniss et al., 2005), but a more appealing solution is to use computer vision. Abstract visual cues have been used in classifying rooms (Tapus and Siegwart, 2006; Ulrich and Nourbakhsh, 2000) and recently representations based on visual object recognition have been presented (Vasudevan et al., 2006) for spatial descriptions.

Apart from semantic labeling, visual information can also be used for map building. In (Se et al., 2002) a 3D representation is built, where locations of distinctive features are reconstructed. Recently we have presented an approach where the environment map is not a 3D reconstruction but is represented as an "appearance graph": a topological representation where nodes represent omnidirectional images taken by the robot and edges are defined by similarities between these images (Zivkovic et al., 2005). We showed that this representation can be used for path planning (Bakker et al., 2005) and for finding a categorical representation (Zivkovic et al., 2006). In this paper we will show that this representation can be used for navigation.

Strategies for mobile robot navigation with omnidirectional vision systems have been reported earlier. (Mariottini et al., 2006) shows the epipolar constraint can be used for moving from one pose to another in a simulator. Navigation over longer paths has been reported by (Argyros et al., 2005), but this system is restricted to travel only along prerecorded trajectories.

In this paper we present visual navigation using the appearance graph, making it possible to drive trajectories not driven in the training phase. In Section B.2 we will first summarize our appearance based topological representation. Then we will explain our navigation strategy, in which we use the epipolar constraint and the constraint of moving over a planar ground floor to obtain a robust heading estimation (Section B.3). In Section B.4 we explain how to navigate over the graph representation. Experiments on orientation estimation and navigation using real data are reported in Section B.5.

B.2 Appearance based topological mapping

In this section we describe the method used for constructing the appearance based map. This has already been reported in (Zivkovic et al., 2005). The goal is to construct a weighted graph $G = (V, S)$ in which the nodes V denote images taken at certain positions in the environment and each link S_{ij} in the graph denotes that image i and j look similar and are thus likely to be taken from more or less the same position (Schaffalitzky and Zisserman, 2002). The similarity measure we use is directly linked to the ability to perform navigation between the two positions. If we can robustly reconstruct the local geometry given the two images then we define a link $S_{ij} > 0$ between the two nodes. The robustness of the reconstruction is expressed in the value of S_{ij} of the link.

We start with a set of images taken by the robot while it was driven around in the environment. In order to get a large overlap in the images the robot is equipped with an omnidirectional vision system consisting of a hyperbolic mirror and an ordinary camera (see (Zivkovic and Booij, 2005) for details). From each of the resulting panoramic images a set of SIFT features is extracted (Lowe, 2004). Then for each pair of feature sets, corresponding points are found by comparing their SIFT descriptors, see Figure B.1. The epipolar geometry is determined using robust estimation techniques, which are also used for robot navigation (more about this in Section B.3). One of the outputs of the estimation method is the number of point correspondences that agree with the epipolar constraint. However from these correspondences there is still a percentage of false feature matches. The number of these false matches is in the order of the number of features found in the two images. By dividing the number of constrained point correspondences by the lowest number of features found in the two images we obtain a similarity measure between 0 and 1. If this value is larger than a certain threshold, which indicates the robustness of the local geometry estimation, the images match and a link S_{ij} is created between the two nodes representing the images with its value set to the similarity.

In Figure B.5 an appearance based map is shown as constructed for the navigation

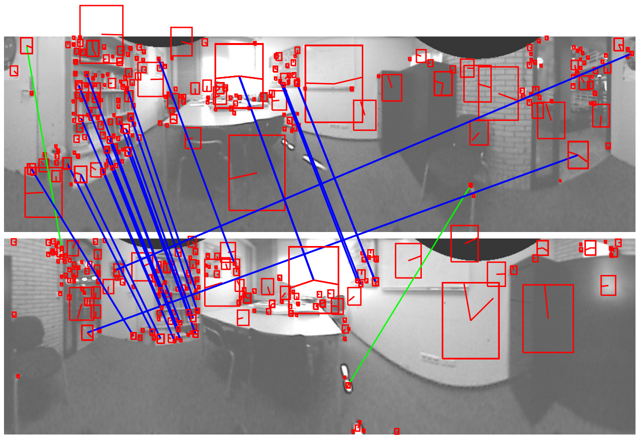


Figure B.1: Matching two images. The red boxes indicate the SIFT features found in the images. The lines connecting two of these features indicate that they correspond. If the line is blue, this means the corresponding pair agrees with the epipolar constraint. If it is green, it does not agree with the epipolar constraint and is thus probably an outlier.

experiments. The map is purely topological, as there is no explicit distance or scale information present in the graph (note that the position of the nodes is based on odometry information, however this is only used for visualizing the graph.) What is contained in the map is information on the neighborhood relation of different parts of the environment. The graph representation is well suited for further processing. In (Zivkovic et al., 2005) we explain how graph clustering techniques are used to find convex spaces in the map, which correspond to rooms and corridors in the environment. Augmenting the graph-clusters results in a semantically labeled map, which can be used for human robot interaction (Spexard et al., 2006).

B.3 Heading estimation using the epipolar geometry

Epipolar geometry estimation is thoroughly discussed in computer vision literature and some standard implementations are readily available (Hartley and Zisserman, 2003; Torr and Murray, 1997). However, because we use an omnidirectional vision system and a robot to obtain the images, there are some special issues to take into account.

We assume that we extracted a set of N matching point pairs from the two panoramic images. The image points are then projected on a sphere around the optical centers with distance 1. Let us denote the 3D points in the current image as $\{\mathbf{x}_1^{(1)}, ..., \mathbf{x}_1^{(N)}\}$, and the corresponding points in the target image as $\{\mathbf{x}_2^{(1)}, ..., \mathbf{x}_2^{(N)}\}$. Omnidirectional vision systems are by default calibrated in order to produce single viewpoint images. So we can use the essential matrix E, instead of the more general fundamental matrix for uncalibrated images, to relate point correspondences in the following way:

$$(\mathbf{x}_1^{(i)})^T E \mathbf{x}_2^{(i)} = 0 \qquad \text{for all } i. \tag{B.1}$$

Using 8 point pairs we can linearly solve E with the 8-point algorithm (Hartley and Zisserman, 2003). However the robot is driving over the planar ground floor. Hence, it can be assumed that the positions of the images do not differ in height and the relative rotation only occurs around the vertical axis. This prior knowledge can be incorporated by restricting the essential matrix in the following form (Brooks et al., 1998; Kosecká et al., 2005):[1]

$$E = \begin{bmatrix} 0 & e(2) & 0 \\ e(4) & 0 & e(6) \\ 0 & e(8) & 0 \end{bmatrix} \tag{B.2}$$

The minimal number of point pairs for the linear estimation of this constrained essential matrix is only[2] 4 and because the solution space is smaller the estimator is less effected by noise.

The essential matrix bears the relative rotation R and translation $\vec{t}$ up to an unknown scale between the positions of the two images as follows:

$$E = RS, \tag{B.3}$$

where S is a skew-symmetric matrix composed of the elements of $\vec{t}$. The essential matrix can be decomposed into 4 different solutions of $\vec{t}$ and R. By imposing the constraint that world points should lie in front of the image surface on which it is projected, we can choose the correct solution (Horn, 1990). The world points that were projected behind one of the image surfaces given the correct R and $\vec{t}$ were obviously produced by false matching. For panoramic images the probability that a false match is in front of both image surfaces is small, because omnidirectional vision systems look in every direction, see Figure B.2. We use this knowledge in the robust estimation process described below.

Generally image points are not noise-free and part of the point pairs found by the matching algorithm is the result of false matching. Therefore a simple least squares method to fit an essential matrix to the data will fail miserably. A fast and robust estimation method that can cope with a large percentage of these false matches is RANSAC (random sample consensus), which we use to determine correct point pairs (Hartley and Zisserman, 2003; Torr and Murray, 1997). RANSAC estimates a large number of essential matrices and chooses that E that agrees with the most point pairs. In each run it first estimates E given 4 randomly chosen point correspondences using the planar version of the 8-point algorithm. Then we check if the four correspondences all lie in front of both image planes. If not, then we discard that estimate of E (this type of model checking was first proposed in (Chum et al., 2004)). If the correspondences are consistent, we use E to reproject all point correspondences of the image pair and count the number inliers. An

[1]Note, that this equation differs from Equation 3.38. This is not a fundamental difference but is caused by a difference in the coordinate system. More specifically, the second and third coordinates are swapped.

[2]As discovered later, the minimal number of required point correspondences is actually 3. This is explained in Section 3.4.3.

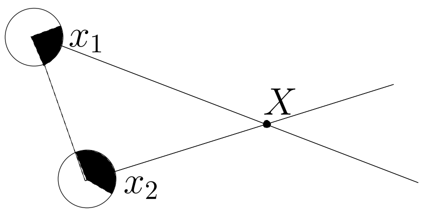


Figure B.2: A 2D visualization of a corresponding point in two panoramic images. The circles denote the panoramic images and the black dot is the worldpoint X. The solid lines are the oriented rays going through the optical center and the worldpoint. The filled parts of the circles denote those places in the images where the oriented ray of the other camera can be projected, i.e. the epipolar line. As can be seen in only a small part, less than 50% of the image, a corresponding point can be found.

inlier in our case, is a correspondence that has a low reprojection error and lies in front of both cameras.

After choosing the run with the highest number of inliers, a final E is computed by taking into account all its inliers. From this E, the R and $\vec{t}$ (up to a scale factor) are determined between the image locations (Horn, 1990). In the remainder of the paper we assume that, if R and $\vec{t}$ can be determined robustly, the robot can indeed move from one location to another. This need not be true, for example if there are obstacles with very few localized features, or if the features are located in a restricted region of the images. In our experimental setup we therefore had a local obstacle algorithm operational using sonar. In the runs we present in this paper we did not need the obstacle avoidance.

The heading ϕ the robot has to drive when navigating from the current image to the target image can be calculated using

$$\phi = \text{atan2}(t_y, t_x). \tag{B.4}$$

B.4 Navigation over the topological map

In this section we describe the framework to navigate to a goal location in the environment mapped by the appearance based graph given the heading estimation explained in section B.3. The general aim is that the robot should be able to navigate to any room in a building by giving it a node in the graph. In our specific case we desire that the robot is able to match the last observation with that of the goal node.

A challenge is that there is no positional information stored in the representation. Thus, two images whose nodes are neighbors in the graph, could have been shot at any distance from each other. Techniques exist to estimate the distance from two images. However these techniques would require us to make an assumption on the position of landmarks in the world, making the system less flexible. Another solution is to find corresponding features in three or more images, which is quite common in the field of visual servoing (see for example (Mezouar et al., 2004)). However, in dynamic environments it will be more difficult to find stable correspondences in three images than in just two.

We take it that the goal location is given by a node in the graph. First Dijkstra's shortest path algorithm (Dijkstra, 1959) is used to compute the distance D_i from every node i in the graph to this goal node. This algorithm requires the links of the graph to be labeled with a distance measure while we have a similarity measure. We choose to define the distance d as $d_{ij} = \frac{1}{S_{ij}}$. The distances of the nodes to goal node are used during driving as a heuristic to drive in the direction of the goal node.

Note that the algorithm in our case gives the shortest path in the appearance space, which is not necessarily the shortest path in the metric space. Because of our distance measure, a shortest path will favor a selection of image sequences which have many features in common. This may imply that the robot will avoid path elements where the local features change rapidly (close to narrow throughways) and will prefer to navigate in the center of large open spaces. We plan to design experiments to test this in more detail: in this paper we focus on the robustness with respect to occlusions.

The navigation procedure directs the robot to one node at a time that can be seen as a subgoal node on a path to the goal node. This path of nodes could have been planned in advance. However this would result in a very inflexible trajectory which would be difficult to traverse in a dynamic environment. In the following we explain how the subgoal nodes are determined dynamically while driving.

At the start of the trajectory the robot localizes itself in the appearance based graph by taking a new observation and comparing it with all the images in the graph following the same matching procedure as used for constructing the graph (see Section B.2). The node of the graph with the highest similarity is chosen as the current subgoal node c of the robot. This procedure is linear in the number of nodes and could thus be time consuming.

If a subgoal node is determined the robot tries to pick a new subgoal by comparing the newest observation with all the neighbors of node c that have a smaller distance D_c to the goal node. If one of these images matches, it becomes the new current subgoal c. This procedure is repeated for the neighbors of the new c, until the node is found that is closest to the goal node and still robustly matches the new observation.

When a subgoal is determined, the heading is estimated in order to drive in its direction. This heading will not be perfectly directed toward the subgoal, partly because of sensor-noise, but also because the environment could have changed after the appearance based map was constructed. Therefore a recency weighted averaging filter is used which to takes into account previous estimates of ϕ.

This smoothed heading is now used to move the robot. It then takes a new observation while driving and repeats the whole procedure. This goes on until the subgoal is equal to the global goal and the robot is stopped, completing the navigation.

We also need some recovery method in case the robot gets lost. It could happen that the robot is repeatedly unable to estimate the heading with the current subgoal, because it finds fewer than[3] 4 corresponding image points. This can be due to changing environmental conditions, but can also be caused by bad heading estimates for the previous observations. If the robot can not find a heading for 10 observations in a row it will try

[3]This should be 3 correspondences, see Footnote 2.

to relocalize itself in the map and start with the new node as subgoal.

B.5 Experiments

For the experiments a Nomad Super Scout II is used which is equipped with an omnidirectional vision system consisting of a hyperbolic mirror and an ordinary camera. The navigation procedures are tested in an office environment.

We first test if the heading estimation works properly by comparing it with ground truth positioning data of the robot. Then a large appearance map is constructed by driving the robot manually through the environment. This map is used for the navigation experiments, in which we compare the length of the traversed path to that of a manually driven path. Also we test the robustness against noise, by obstructing part of the view of the robot while it is driving.

B.5.1 Heading estimation

First the low level heading estimation is tested by comparing it with the ground truth positions and orientations of the robot. A small data set is taken by the robot on a 3 by 3 grid of approximately 2 square meters in size. On each point of the grid 4 images are taken with the robot facing in 4 different directions, giving a total of 36 images.

For each pair of images the heading is computed given the method explained in sections B.2 and B.3. The headings between images taken at the same location are meaningless and thus ignored. The estimated heading is compared with the ground truth heading calculated on the basis of odometry information, which is quite accurate at such small distances. In Figure B.3 a histogram is plotted of the difference between estimated and the ground truth heading. The standard deviation of the error is 0.31 radians. Although the images were less than 3 meters apart, the results indicate what we can expect of the heading estimation during navigation.

B.5.2 Appearance based mapping

The robot was driven manually through the environment, consisting of a U-shaped hallway and 3 rooms. While the robot was driving images were taken at a rate of 1 image per second. In Figure B.4 the approximate positions of the images are shown. The position of these images were derived using the odometry information of the robot. Errors in the odometry were corrected somewhat to make the visualization more clear. This is also used for the figure showing the graph. We must stress that the odometry errors did not influence the outcome of the navigation, as we do not use the odometry readings in our methods.

An appearance based topological map is constructed using the images as described in section B.2, see Figure B.5 for the result. The value that is used to threshold the similarity value is set to 0.05, which seemed to work well for a different dataset taken in another environment. This basically means that 5 out of the 100 image features should have a corresponding feature in the other image, which is constrained by the epipolar geometry.

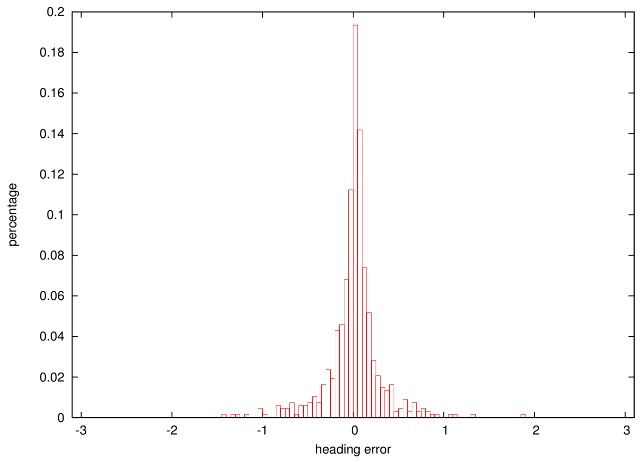


Figure B.3: Histogram of the differences between the estimated and the ground truth heading of all pairs of images.

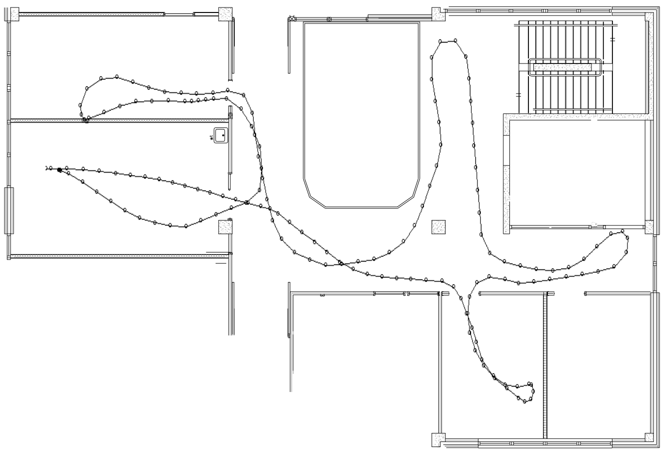


Figure B.4: The path the robot followed while it was manually driven through the environment. The robot started at the lower left of the figure and drove towards the room on the lower right. The circles denote the positions at which panoramic images were taken.

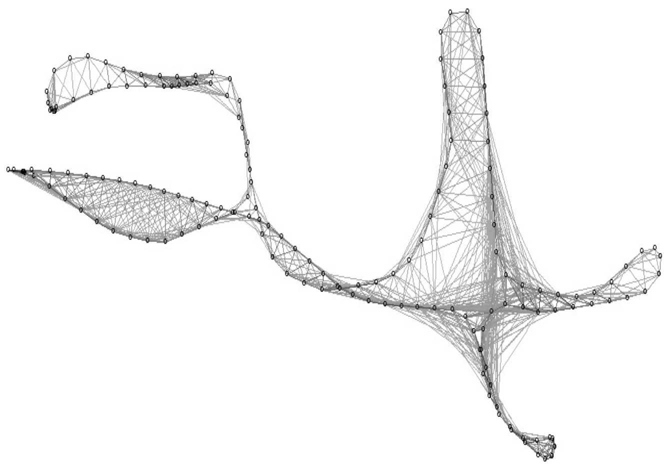


Figure B.5: The appearance based graph. The circles again denote the approximate image positions and the lines connecting them indicate matching images. The gray-value of the lines correspond with the similarity value of the match.

As can be seen in the figure no links were created between images that were taken from very different locations. The matching method thus shows to be robust against similar looking but different office rooms.

B.5.3 Robot navigation

The robot is put in a position in the mapped environment and a goal node is picked from the graph in another part of the environment. This is repeated two times creating two start and end positions for which the robot should find a path and navigate over it. We let the robot navigate 3 times over both paths.

All 6 runs were completed successfully, without having to use the recovery method. In two occasions a heading to the subgoal could not be calculated. However this did not cause the robot to loose track of his path. The robot drove smoothly to the goal node stopping in its vicinity. In Figure B.6 two of the traversed paths are shown. The other 4 paths were very similar to these ones. As can be seen the robot did not drive a the trajectory that was driven while taking the dataset. Rather, it used a path of nodes that was much shorter.

Quantitatively evaluating the performance of robot navigation is not a straightforward task. It is common to report the metric error of the final robot position given an exact goal position (Argyros et al., 2005; Mariottini et al., 2006). However this error depends solely on the last stages of the navigation task, which is only interesting if the start and goal position lie close together. It seems more important to measure if the robot “takes wrong turns” while driving through the environment, from which it has to back up. This would have a great impact on the length of the path the robot traversed. In table B.1 the average path length and the standard deviation is given. For comparison the robot

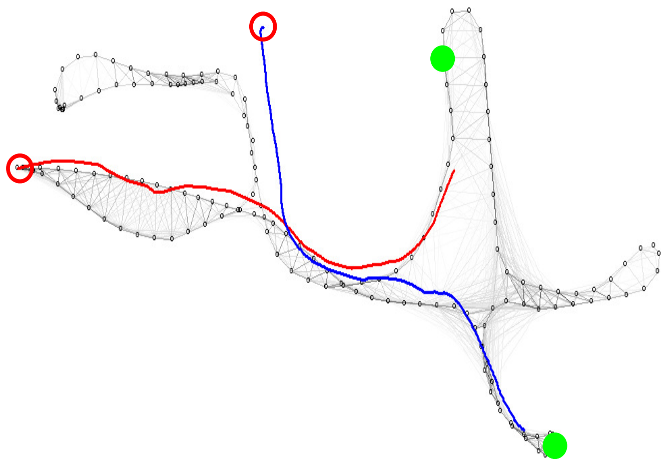


Figure B.6: Two of the traversed paths depicted by the thick blue and red line visualized on top of the graph. As can be seen the paths were quite smooth. The fact that the upper left part of the blue path does not lie on the graph is probably a result of bad odometry readings.

Table B.1: Average driven path lengths in meters $\pm$ the standard deviation for autonomous and manual navigation

	path 1	**path 2**
Auto	$13.8 \pm .4$	$12.4 \pm .8$
Manual	$14.2 \pm .3$	$12.1 \pm .3$

was also driven manually from the start positions to the positions where the robot had stopped, by an experienced user. This is also repeated 3 times per path. The lengths of the manually driven paths are comparable with those of the autonomously driven paths, indicating that the robot did follow a correct path to the goal position (see table B.1).

B.5.4 Navigation with visual occlusions

To put more strain on the visual navigation method we now test the ability to drive while part of the view is blocked by people walking next to and in front of the robot. See Figure B.7 for an indication of the view the robot has while 4 persons are standing next to it. The persons are walking very near the robot at approximately 20 cm distance. The path that had to be traversed is the same as one of the paths in the previous section. Tests are conducted with respectively 1, 2, 3 and 4 persons.

The robot still managed to reach the goal location in all 4 tests. Nonetheless it was clear that for every person that was added, the navigation was a bit more difficult. In table B.2 it is shown that the path length increases if a larger part of the view is blocked.

Table B.2: Path lengths in meters with people blocking the view

#Persons	**path lengths**
0	14.2
1	15.2
2	18.0
3	19.0
4	23.6

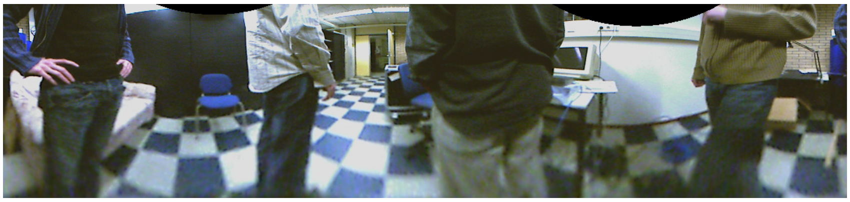


Figure B.7: Four persons blocking the view of the robot.

This is not only caused by small divergences of the correct path, but also because the robot sometimes took a longer route around the pillar in the hallway. Surprisingly the robot never had to use its recovery method. The number of times that the heading to the subgoal could not be estimated did increase though. For one and two persons it could still match 100% of the observations, but this decreased to 90% for the runs with three person and four persons.

During the test with 4 persons an additional thing happened. Because no collision avoidance was used and the robot was sometimes heading for a doorpost or the pillar, we had to stop it manually and push it back. This happened 3 times.

B.6 Conclusion

We presented a navigation system that can use an appearance based topological map as its representation of the environment. The robot is able to find and traverse paths in the visual domain and can navigate from one state to the other. The found paths do not differ significantly with paths taken by a human, when comparing the path lengths.

Navigation proved to be robust in a dynamic environment with people walking close to the robot. Our navigation system is based on a search for a path given all the images available. This in contrast to existing systems for visual navigation, which drive over predefined paths of images, which were learnt during the exploration phase or even hard coded in a map given to the robot. Our approach is robust against changes in the environment and persons or objects blocking certain paths.

Note that our navigation system is based only on an omnidirectional vision system. This same sensor can be used for a range of other tasks such as object or person detection. Also the appearance based map we use for navigation, is used in other work for localization and conceptualization, splitting the map into rooms, corridors, etc.

Currently we are integrating the navigation approach in a more complete robot system that incorporates people detection, people following and exploration. All these methods make use of the same omnidirectional vision system.

C Moving from augmented to interactive mapping

Topological view-based maps are very suitable for adding semantic information such as room labels. Here we show that a real time view-based mapping system allows a robot to interactively add and disambiguate semantic information while driving around and building the map. The system is demonstrated in a small real home environment. In order to perform mapping in real time the Connected Dominating Set based mapping procedure was used as explained in Chapter 5.

This appendix was presented earlier as a paper in the Workshop Interactive Robot Learning at the Robotics: Science and Systems conference in 2008 and was published in its proceedings (Booij et al., 2008). It contains some overlap with the rest of the thesis.

C.1 Introduction and problem statement

Recently there has been a growing interest in human augmented mapping (A. Diosi and Kleeman, 2005; Spexard et al., 2006). That is: a mobile robot builds a low level spatial representation of the environment based on its sensor readings while a human provides labels for human concepts, such as rooms, which are then augmented or anchored to this representation or map (Saffiotti and LeBlanc, 2000). Given such an augmented map the robot has the ability to communicate with the human about spatial concepts using the labels that the human understand. For instance, the robot could report it is in the "Kitchen", instead of a set Cartesian coordinates which are probably meaningless to the human.

Even if the underlying mapping method is perfect, two main problems occur in the approach of augmented mapping. When guiding a robot through a number of rooms, humans tend to not provide labels for every visited room (Topp et al., 2006). The result is that the robot has difficulty to model where one room ends and the other room starts. This problem could be solved by detecting room transitions through the sensor data. Although good attempts using such an approach have been made in office environments (Zender et al., 2008; Martínez-Mozos et al., 2007), applying these to other environments such as real homes is nontrivial. Another problem is that the generalization of the labeled map to newly acquired sensor data can be much different from the humans ideas. That is: there is a mismatch between the human representation and the representation of the robot. In our case the robots generalizes labels using visual similarities, while humans could use the function of the room. Even among humans there are differences between spatial representations. Think of a living room with an open kitchen. Where does the living room end and the kitchen begin?

Our solution to both of these problems is to use pro-active human robot interaction. We

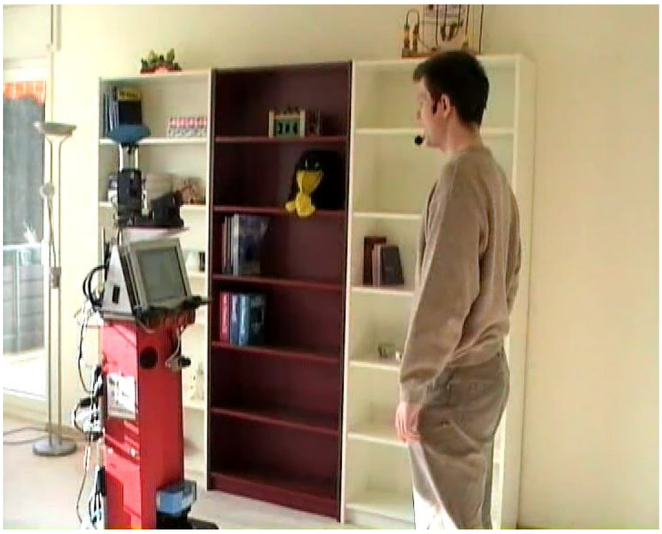


Figure C.1: Biron and human guide in a home environment.

briefly describe how the robot learns a map of the environment using a vision sensor and active dialog with a human guide. The method is implemented on Biron (the Bielefeld Robot Companion), see Figure C.1, which supports an integrated human robot interaction system based on XCF (XML Communication Framework) complete with person attention, spoken dialog, person following, gesture recognition and localization components (Fritsch et al., 2005).

C.2 Augmented mapping

C.2.1 Appearance based topological mapping

To map the environment we take images with an omnidirectional vision system. From each image SIFT features are extracted which are used to find image point correspondences between pairs of images by matching their SIFT descriptors. False point correspondences are then removed by imposing the epipolar constraint. We define a distance measure between two images i and j by:

$$d_{ij} = \frac{\min(\text{\#SIFTS}_i, \text{\#SIFTS}_j)}{\text{\#correspondences}_{ij}},$$

where #SIFTS$_i$ denotes the number of SIFT features extracted from an image i, and #correspondences$_{ij}$ denotes the number of correspondending features of images i and j, that are constrained by an epipolar geometry.

These computed distances are put in a graph representation in which the nodes denote the images and distances are put on the links, effectively creating a topological map of

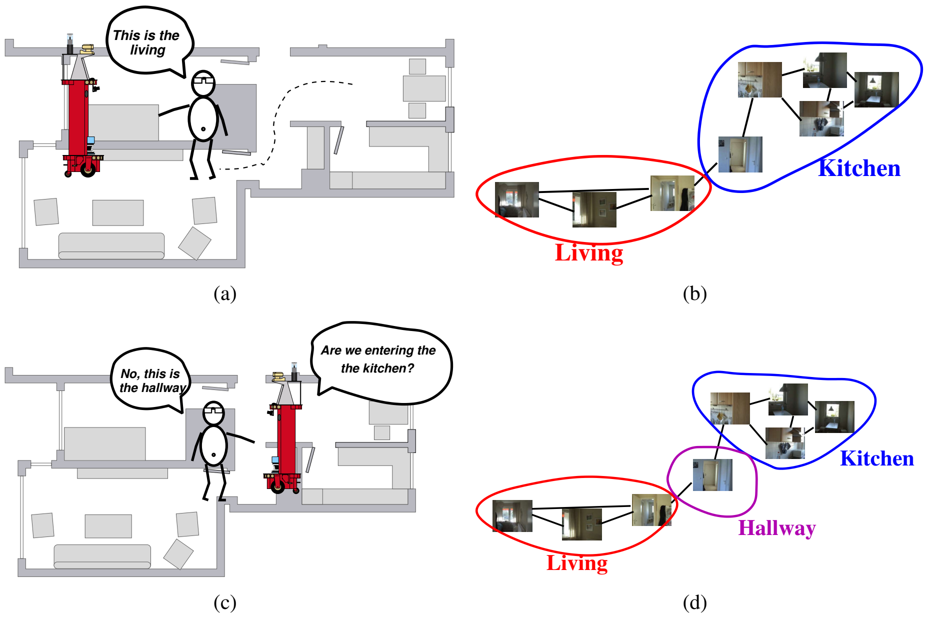


Figure C.2: A sketch of the proposed method. (a) The human guide provides a label. (b) After a second label is provided the map consists of two subgraphs. (c) The robot reports a room transition on which the human provides feedback. (d) The feedback is used to update the map.

the environment. If the distance is above a certain threshold (which was set to 10 in our experiments) then no link was created.

The complete map building system runs in real time on one of the robot-laptops, processing about one new image per second. To keep the number of comparisons limited we used the Connected Dominating Set method to pick key images from the previous image set. For an in depth treatment of this map building scheme see (Booij et al., 2006).

C.2.2 Human augmentation of room labels

While the robot is driving through the environment following the human guide and building a topological map, room-labels can be provided to the robot, see Figure C.2.2 for an example. This is performed by commanding the robot to stop and telling the robot the name of the room it is in, e.g. “This is the kitchen” (see Figure C.2(a)). To handle miscommunication, a powerful grounding-based dialog system is used that can handle complex conversational repair behavior and facilitate a smooth conversation (see (Spexard et al., 2006) for more information). The given label is then added to the next node (image) that is added to the map.

Using the given labels and the structure of the graph the robot can partition the map into different subgraphs. Every node is assigned to that label corresponding to the closest labeled node computed with Dijkstra's shorter path algorithm (Dijkstra, 1959) (see Figure C.2(b)). Effectively we are exploiting here the fact that images taken in a convex space, which usually correspond to the notion of rooms, are visually much more similar than images taken while the robot moved through a narrow passage, a door.

C.3 Interactive mapping

As can be seen in Figure C.2(b) the transition from the "Living room" to the "Dining room" is probably not learned in the way the human had in mind when giving the labels. The human would probably not notice this until it would send the robot to the "Living room" after which the robot would move to the hallway. This can easily be solved by making the robot pro-actively interact with the human.

Every time the robot adds a new image to the map it computes its corresponding label. If this label is different than the label of the previously added node, the robot reports this to the human in the form of a question. In the case of Figure C.2(c) the robot asked "We just entered the living room, right?". The human now has the opportunity to provide feedback, possibly reducing the mismatch with his/her own representation, see Figure C.2(d). If later the robot would really enter the "Living room" it will again report this to the human confirming that it has correctly learned the transition.

A technical detail is that the robot does not stop driving while reporting room change to the human, so as to not interrupt the tour. Thus new nodes are added to the graph while it awaits an answer. The possibly corrected label is put on the node which triggered the robot. This could lead to race conditions if there are a lot of transitions close to each other, e.g. if different locations in the room are also labeled. In the conducted experiments, however, we did not experience such problems.

C.4 Results

The new interactive mapping approach was recently implemented on the Biron robot. First test trials were performed in a rented apartment at Bielefeld which was furnished to look like a real home environment. See `http://www.science.uva.nl/~obooij/research/mappingHRI/index.html` which features a video shot during one of the trials illustrating the capabilities of the complete interactive mapping system.

The robot captured panoramic images once every 2 seconds and the tour took around 5 minutes resulting in a total set of 158 images. The complete mapping system, including the image processing, is performed during the tour in real-time on one of the laptops attached to the robot.

In Figures C.3(a)-(e) the spatial representation is plotted using hand-corrected odometry data. Note, however, that this odometry data was not used by the mapping algorithm.

In Figure C.3(a) the robot drove from the living room at the bottom right of the figure through the hallway to the kitchen on the upper left. By then the only label that was given was in the living room, so it groups every new node with that label. In Figure C.3(b) it is provided a new label "Dining room" and as can be seen the graph is split into two groups according to their distance over the graph. The cut between these two groups is located somewhere inside the small hallway.

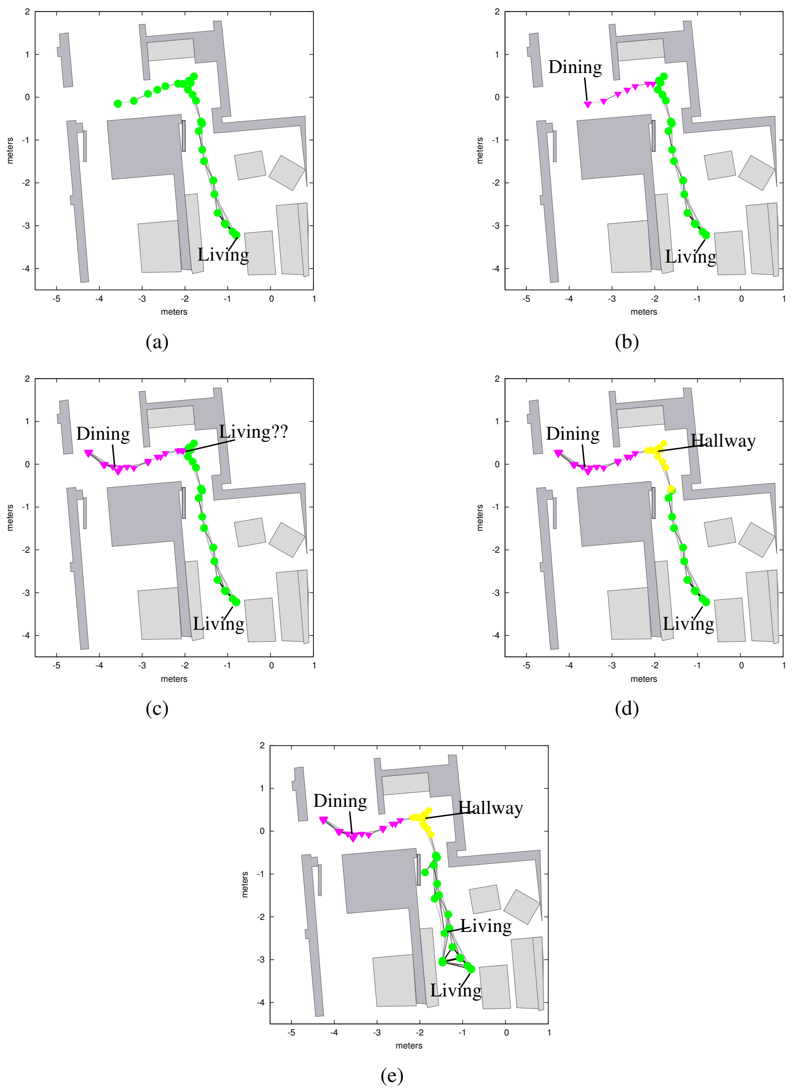


Figure C.3: The spatial representation built by the robot. The different symbols denote nodes (images) of the graph. The lines between the symbols denote links between the lines, with darker colored lines representing links with a smaller distance. Circles denote nodes belonging to the “Living room”, squares to the “Dining room” and pentagons to the small “Hallway”. Symbols linked with a label represent nodes that were labeled by the guide. In addition part of the ground-truth floor map is plotted on top for reference.

This became apparent to the guide in Figure C.3(c) where the robot was guided back to the hallway. The robot proactively starts a dialog by asking the guide “Did we just enter the Living Room?”. The guide can then correct the robot by giving it an other existing or new label. In the experiment the guide gives the new label “Hallway”. This new label is added to the map, splitting the graph in three parts, see Figure C.3(d). After reentering the living room the robot again asked if this was the “Living room” which was confirmed by the guide resulting in another node being labeled. In Figure C.3(e) the final spatial representation is shown as built by the robot.

C.5 Conclusion

We have shown that using relatively simple human robot interaction techniques we can solve two problems apparent in augmented mapping systems. The robot actively asks the labels of rooms that were not labeled at the first visit and decreases the mismatch between the human representation of room transitions and the robots representation. The complete system can be run in real time on a laptop and has been shown to work in a real home environment.

Future work is directed to gathering larger evidence for the feasibility of the interactive localization approach. The system scales well to larger environments and is flexible because it uses only a vision sensor.

List of publications

The research described in this thesis has resulted in the following publications:

O. Booij, Z. Zivkovic and B. Kröse. Efficient probabilistic planar robot motion estimation given pairs of images. In *Proceedings of Robotics: Science and Systems*, 2010. Chapter 3.

O. Booij, Z. Zivkovic and B. Kröse. Efficient data association for view based SLAM using connected dominating sets. In *Robotics and Autonomous Systems*, 57(12):1225-1234, Dec. 2009. Chapter 5.

Z. Zivkovic, O. Booij and B. Kröse, E. Topp, H.I. Christensen. From sensors to human spatial concepts: an annotated dataset. In *IEEE Transactions on Robotics*, 24(2):501-505, Apr. 2008. Appendix A.

O. Booij, B. Kröse, J. Peltason, T. Spexard and M. Hanheide. Moving from augmented to interactive mapping. In *Proceedings of the Interactive Robot Learning Workshop during the Robotics: Science and Systems Conference (RSS)*, Jun. 2008. Appendix C.

O. Booij, Z. Zivkovic and B. Kröse. Sampling in image space for vision based SLAM. In *Proceedings of the Inside Data Association Workshop during the Robotics: Science and Systems Conference (RSS)*, June 2008. Chapter 5.

B. Kröse, O. Booij and Z. Zivkovic. A geometrically constrained image similarity measure for visual mapping, localization and navigation. In *Proceedings of the 3rd European Conference on Mobile Robots*, 2007. Chapters 3, 4 and Appendix B.

O. Booij, B. Terwijn, Z. Zivkovic and B. Kröse. Navigation using an appearance based topological map. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2007. Appendix B.

T. Spexard, S. Li, B. Wrede, J. Fritsch, G. Sagerer, O. Booij, Z. Zivkovic, B. Terwijn and B. Kröse. BIRON, where are you? Enabling a robot to learn new places in a real home environment by integrating spoken dialog and visual localization. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006. Chapters 3 and 5.

O. Booij, Z. Zivkovic and B. Kröse. Sparse appearance based modeling for robot localization. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006. Chapter 5.

Other recent publications:

I. Esteban, O. Booij, J. Dijk and F. Groen. On the bending problem for large scale mapping. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

I. Esteban and O. Booij and Z. Zivkovic and B. Kröse. Mapping Large Environments with an Omnivideo Camera. In the *Workshop Proceedings of the Conference On Simulation, Modeling and Programming for Autonomous Robots*, 2008.

Z. Zivkovic, O. Booij and B. Kröse, From images to rooms, IEEE Transactions on Robotics, 55(5):411-418, May, 2007.

P. T. Quinlan, H. L. J. van der Maas, B. R. J. Jansen, O. Booij and M. Rendell. Rethinking stages of cognitive development: An appraisal of connectionist models of the balance scale task. In *Elsevier Cognition*, 103(3):413-459, Jun. 2007.

O. Booij and H. T. Nguyen. A gradient descent rule for spiking neurons emitting multiple spikes. In *Information Processing Letters*, 95(6):552-558, Sep. 2005.

Dankwoord[1]

Er zijn een aantal mensen die hebben bijgedragen aan het verschijnen van dit proefschrift. Zonder hen was ik allang gestopt met mijn AiO-schap om iets veel leukers te doen.

Allereerst mijn begeleiders Ben en Zoran. Zoran heeft me geleerd dat technische kennis en wetenschappelijk inzicht samen leiden tot praktisch oplossingen. Zijn pragmatische en efficiënte aanpak inspireert me. Ben heeft me geleerd hoe over die gevonden oplossingen te publiceren. Ik waardeer ook zijn directe manier van het benaderen van mensen (hij leerde me dat wetenschappers ook gewoon een telefoon hebben). Vooral de laatste jaren heeft hij flink wat bergen moeten verzetten om mij zo ver te krijgen mijn proefschrift af te ronden. Mijn dank hiervoor.

Het samenwerken met IASers en ook ISISers was leuk en leerzaam. Het was Bas die er voor zorgde dat mijn software kon draaien op diverse robots. Eigenlijk was hij de enige die daadwerkelijk iets aan intergratie deed in het "Intergrating Project" Cogniron. De datasets die ik heb gebruikt voor dit proefschrift zijn voor een groot deel door hem tot stand gekomen. Mijn kamergenoten (Carsten, Michael, Thanasis en vast nog iemand die ik vergeet) zorgden voor de nodige afleiding en ook voor het delen van proefschriftleed. Doordat ik hen zag ploeteren, dacht ik dat het erbij hoorde, bedankt.

En dan een dankbaarheid van een hele andere orde. In de maand dat ik begon aan mijn AiO-schap bleek Jurny zwanger te zijn. Een kleine negen maanden later met Koen erbij zag ons leven er opeens heel anders uit. Ik leerde dat het eigenlijk niet kan: jonge kinderen en AiO-schap. Jurny heeft haar ambities opzij gezet, om er voor te zorgen dat ik verder kon werken aan mijn proefschrift. Ze heeft vaak voor zowel moeder als vader gespeeld, wanneer ik mijn roes lag uit te slapen na een nachtje doorwerken, of ergens bij een conferentie aan het feesten was. Ik zal het, na vandaag, nooit meer doen.

[1]In dutch

Bibliography

A. C. Murillo, J. J. G. and Sagues, C. (2008). *Topological and metric robot localization through computer vision techniques - book "Unifying Perspectives in Computational and Robot Vision"*, pages 113–127. Lecture Notes Electrical Engineering. Springer Lecture Notes Electrical Engineering Series.

A. Diosi, G. T. and Kleeman, L. (2005). Interactive slam using laser and advanced sonar. In *2005 IEEE International Conf. on Robotics and Automation, ICRA 2005*, Barcelona, Spain.

Anati, R. and Daniilidis, K. (2009). Constructing topological maps using markov random fields and loop-closure detection. In Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C. K. I., and Culotta, A., editors, *Advances in Neural Information Processing Systems 22*, pages 37–45. MIT Press.

Andreasson, H., Duckett, T., and Lilienthal, A. J. (2008). A minimalistic approach to appearance based visual slam. *Transactions on Robotics*, 24(6):991–1001.

Argyros, A. A., Bekris, K. E., Orfanoudakis, S. C., and Kavraki, L. E. (2005). Robot homing by exploiting panoramic vision. *Autonomous Robots*, 19(1):7–25.

Armangué, X. and Salvi, J. (2003). Overall view regarding fundamental matrix estimation. *Image Vision Comput.*, 21(2):205–220.

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., and Wu, A. Y. (1998). An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45(6):891–923.

Bailey, T. and Durrant-Whyte, H. (2006). Simultaneous localisation and mapping (SLAM): Part II State of the art. *Robotics and Automation Magazine*, 13(3):108–117.

Bakker, B., Zivkovic, Z., and Kröse, B. (2005). Hierarchical dynamic programming for robot path planning. In *IROS*, pages 3720–3625, Edmonton, Canada. IEEE/RSJ.

Basri, R., Rivlin, E., and Shimshoni, I. (1999). Visual homing: Surfing on the epipoles. *International Journal of Computer Vision*, 33(2):117–137.

Bay, H., Tuytelaars, T., and Gool, L. J. V. (2006). SURF: Speeded up robust features. In Leonardis, A., Bischof, H., and Pinz, A., editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 3951 of *Lecture Notes in Computer Science*, pages 404–417, Graz, Austria. Springer.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer.

Booij, O., Kröse, B., Peltason, J., Spexard, T., and Hanheide, M. (2008). Moving from augmented to interactive mapping. In *Proceedings of the Robotics: Science and Systems workshop Interactive Robot Learning*, pages 21–23.

Booij, O., Terwijn, B., Zivkovic, Z., and Kröse, B. (2007). Navigation using an appearance based topological map. In *ICRA*, pages 3927–3932, Roma, Italy. IEEE.

Booij, O., Zivkovic, Z., and Kröse, B. (2006). Sparse appearance based modeling for robot localization. In *IROS*, pages 1510–1515, Beijing, China. IEEE/RSJ.

Booij, O., Zivkovic, Z., and Kröse, B. (2009). Efficient data association for view based slam using connected dominating sets. *Robotics and Autonomous Systems*, 57(12):1225–1234.

Booij, O., Zivkovic, Z., and Kröse, B. (2010). Efficient probabilistic planar robot motion estimation given pairs of images. In *Proceedings of Robotics: Science and Systems (RSS)*, Zaragoza, Spain.

Brooks, M., de Agapito, L., Huynh, D., and Baumela, L. (1998). Towards robust metric reconstruction via a dynamic uncalibrated stereo head. *Image Vision Comput.*, 16(14):989–1002.

Brückner, M., Bajramovic, F., and Denzler, J. (2008). Experimental evaluation of relative pose estimation algorithms. In Ranchordas, A. and Araújo, H., editors, *VISAPP (2)*, pages 431–438. INSTICC - Institute for Systems and Technologies of Information, Control and Communication.

Bunschoten, R. (2003). *Mapping and Localization from a Panoramic Vision Sensor*. PhD thesis, University of Amsterdam.

Callmer, J., Granström, K., Nieto, J., and Ramos, F. (2008). Tree of words for visual loop closure detection in urban SLAM. In Kim, J. and Mahony, R., editors, *Proceedings of the 2008 Australasian Conference on Robotics & Automation*, page 8.

Censi, A. and Carpin, S. (2009). HSM3D: Feature-less global 6DOF scan-matching in the Hough/Radon domain. In *ICRA*, Kobe, Japan. IEEE.

Chum, O. and Matas, J. (2010). Large-scale discovery of spatially related images. *PAMI*.

Chum, O., Werner, T., and Matas, J. (2004). Epipolar geometry estimation via RANSAC benefits from the oriented epipolar constraint. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, volume 1, pages 112–115.

Clemente, L., Davison, A., Reid, I., Neira, J., and Tardós, J. (2007). Mapping large loops with a single hand-held camera. In *Proceedings of Robotics: Science and Systems (RSS)*, Atlanta, USA.

Cummins, M. and Newman, P. (2008). FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. *The International Journal of Robotics Research*, 27(6):647–665.

Cummins, M. and Newman, P. (2009). Highly scalable appearance-only SLAM - FAB-MAP 2.0. In *Proceedings of Robotics: Science and Systems (RSS)*, Seatle, USA.

Datta, R., Joshi, D., Li, J., and Wang, J. Z. (2008). Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2).

Davison, A. J. (1999). *Mobile Robot Navigation Using Active Vision*. PhD thesis, University of Oxford.

Davison, A. J., Reid, I. D., Molton, N., and Stasse, O. (2007). MonoSLAM: Real-time single camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6):1052–1067.

Dellaert, F. and Kaess, M. (2006). Square Root SAM: Simultaneous location and mapping via square root information smoothing. *IJRR*, 25(12):1181. Special issue on RSS 2006.

den Hollander, R. and Hanjalic, A. (2007). A combined RANSAC-Hough transform algorithm for fundamental matrix estimation. In *18th British Machine Vision Conference*. University of Warwick, UK.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.

Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous localisation and mapping (SLAM): Part I The essential algorithms. *Robotics and Automation Magazine*, 13(2):99–108.

Elinas, P. and Little, J. J. (2005). σMCL: Monte-Carlo localization for mobile robots with stereo vision. In *Proceedings of Robotics: Science and Systems (RSS)*, Cambridge, USA.

Esteban, I., Booij, O., Zivkovic, Z., and Kröse, B. (2008). Mapping large environments with an omnivideo camera. In *Workshop Proceedings of the Conf. On Simulation, Modeling and Programming for Autonomous Robots*, pages 297–306, Venice, Italy.

Eustice, R. (2005). *Large-area visually augmented navigation for autonomous underwater vehicles*. PhD thesis, MIT - Woods Hole Oceanographic Institute.

Eustice, R. M., Singh, H., and Leonard, J. J. (2006). Exactly sparse delayed-state filters for view-based SLAM. *IEEE Transactions on Robotics*, 22(6):1100–1114.

Faugeras, O. and Maybank, S. (1990). Motion from point matches: Multiplicity of solutions. *IJCV*, 4(3):225–246.

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Com. of the ACM*, 24(6).

Folkesson, J., Jensfelt, P., and Christensen, H. (2005). Vision SLAM in the measurement subspace. In *ICRA*, pages 30–35, Barcelona, Spain. IEEE.

Franz, M., Schölkopf, B., and Bülthoff, H. (1998). Where did I take that snapshot? Scene-based homing by image matching. *Biological Cybernetics*, 79:191–202.

Fraundorfer, F., Engels, C., and Nister, D. (2007). Topological mapping, localization and navigation using image collections. In *IROS*, pages 3872–3877, San Diego, USA. IEEE/RSJ.

Frese, U. (2006a). A discussion of simultaneous localization and mapping. *Auton. Robots*, 20(1):25–42.

Frese, U. (2006b). Treemap: An o(log n) algorithm for indoor simultaneous localization and mapping. *Auton. Robots*, 21(2):103–122.

Frese, U., Larsson, P., and Duckett, T. (2005). A multilevel relaxation algorithm for simultaneous localisation and mapping. *IEEE Transactions on Robotics*, 21(2):196–207.

Frese, U. and Neira, J. (2009). Editorial: Inside data association. *Robot. Auton. Syst.*, 57(12):1155–1156.

Fritsch, J., Kleinehagenbrock, M., Haasch, A., Wrede, S., and Sagerer, G. (2005). A flexible infrastructure for the development of a robot companion with extensible HRI-capabilities. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 3419–3425, Barcelona, Spain.

Goedemé, T., Tuytelaars, T., Gool, L. V., Vanhooydonck, D., Demeester, E., and Nuttin, M. (2005a). Is structure needed for omnidirectional visual homing? In *Proceedings of the 6th IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 303–308.

Goedemé, T., Tuytelaars, T., Vanacker, G., Nuttin, M., and Gool, L. V. (2005b). Feature based omnidirectional sparse visual path following. In *IROS*, pages 1003–1008, Edmonton, Canada. IEEE/RSJ.

Grisetti, G., Stachniss, C., and Burgard, W. (2009). Non-linear constraint network optimization for efficient map learning. *IEEE Transactions on Intelligent Transportation Systems"*, 10(3):428–439.

Gross, H.-M. and Koenig, A. (2004). Robust omniview-based probabilistic self-localization for mobile robots in large maze-like environments. In *ICPR (3)*, pages 266–269.

Guha, S. and Khuller, S. (1998). Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374–387.

Haasch, A., Hohenner, S., Hüwel, S., Kleinehagenbrock, M., Lang, S., Toptsis, I., Fink, G. A., Fritsch, J., Wrede, B., and Sagerer, G. (2004). Biron – the bielefeld robot companion. In Prassler, E., Lawitzky, G., Fiorini, P., and Hägele, M., editors, *Proc. Int. Workshop on Advances in Service Robotics*, pages 27–32, Stuttgart, Germany. Fraunhofer IRB Verlag, Fraunhofer IRB Verlag.

Ham, J., Lin, Y., , and Lee, D. D. (2005). Learning nonlinear appearance manifolds for robot localization. In *IROS*, Edmonton, Canada. IEEE/RSJ.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *4th Alvey Conference*, pages 147–152, Manchester, UK.

Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision, second edition*. Cambridge University Press.

He, W., Yamashita, T., Lu, H., and Lao, S. (2009). SURF tracking. In *Proceedings of the International Conference on Computer Vision (ICCV)*, Kyoto, Japan. IEEE.

Heeger, D. J. and Jepson, A. D. (1992). Subspace methods for recovering rigid motion i: algorithm and implementation. *Int. J. Comput. Vision*, 7(2):95–117.

Horn, B. K. P. (1990). Relative orientation. *Int. Journal of Computer Vision*, 4(1):59–78.

Hough, P. V. C. (1962). Method and means for recognizing complex patterns. U.S. Patent 3,069,654.

Ishiguro, H. and Tsuji, S. (1996). Image-based memory of environment. In *in Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pages 634–639.

Jogan, M. and Leonardis, A. (2003). Robust localization using an omnidirectional appearance-based subspace model of environment. *Robotics and Autonomous Systems, Elsevier Science*, 45(1):51—72.

J.Shi and J.Malik (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Anlysis and Machine Intelligence*, 22(8):888–904.

Kaess, M., Ranganathan, A., and Dellaert, F. (2008). iSAM: Incremental smoothing and mapping. *IEEE Trans. on Robotics*, 24(6):1365–1378.

Kanatani, K. (1996). *Statistical Optimization for Geometric Computation: Theory and Practice*. Elsevier Science Inc., New York, NY, USA.

Konolige, K. (2005). Slam via variable reduction from constraint maps. In *ICRA*, pages 667–672, Barcelona, Spain. IEEE.

Konolige, K. and Bowman, J. (2009). Towards lifelong visual maps. In *IROS*, St. Louis, USA. IEEE/RSJ.

Konolige, K., Bowman, J., Chen, J. D., Mihelich, P., Calonder, M., Lepetit, V., and Fua, P. (2009). View-based maps. In *Proceedings of Robotics: Science and Systems (RSS)*, Seatle, USA.

Kosecká, J., Li, F., and Yang, X. (2005). Global localization and relative positioning based on scale-invariant keypoints. *Robotics and Autonomous Systems*, 52(1):27–38.

Kröse, B. J. A., Vlassis, N. A., Bunschoten, R., and Motomura, Y. (2001). A probabilistic model for appearance-based robot localization. *Image Vision Comput.*, 19(6):381–391.

Kuipers, B. J. (1978). Modeling spatial knowledge. *Cognitive Science*, 2:129–153.

Lamon, P., Nourbakhsh, I., Jensen, B., and Siegwart, R. (2001). Deriving and matching image fingerprint sequences for mobile robot localization. In *ICRA*, Seoul, Korea.

Li, F. and Kosecká, J. (2006). Probabilistic location recognition using reduced feature set. In *ICRA*, pages 3405–3410, Orlando, Florida, USA. IEEE.

Li, X., Wu, C., Zach, C., Lazebnik, S., and Frahm, J. (2008). Modeling and recognition of landmark image collections using iconic scene graphs. In *ECCV*.

Longuet-Higgins, H. C. (1981). A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2*, pages 1150–1157. IEEE Computer Society.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision*, 60(2):91–110.

Lu, F. and Milios, E. (1997). Globally consistent range scan alignment for environment mapping. *Auton. Robots*, 4(4):333–349.

Mallot, H. A. and Gillner, S. (2000). Route navigating without place recognition: What is recognised in recognition-triggered responses? *Perception*, 29(1):43–55.

Mariottini, G., Oriolo, G., and Prattichizzo, D. (2006). Image-based visual servoing for nonholonomic mobile robots with central catadioptric camera. In *ICRA*, pages 497 – 503, Orlando, Florida.

Mariottini, G. and Prattichizzo, D. (2008). Image-based visual servoing with central catadioptric camera. *International Journal of Robotics Research*, 27:41–57.

Martínez-Mozos, O., Triebela, R., Jensfeltb, P., Rottmanna, A., and Burgarda, W. (2007). Supervised semantic labeling of places using information extracted from sensor data. *Robotics and Autonomous Systems*, 55(5):391–402.

Maybank, S. (1992). *Theory of Reconstruction from Image Motion*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Menegatti, E., Maeda, T., and Ishiguro, H. (2004). Image-based memory for robot navigation using properties of omnidirectional images. *Robotics and Autonomous Systems*, 47(4):251–267.

Mezouar, Y., Hadj Abdelkader, H., Martinet, P., and Chaumette, F. (2004). Visual servoing from 3d straight lines with central catadioptric cameras. In *Fifth Workshop on Omnidirectional Vision, Omnivis'2004*, Prague, Czech Republic.

Milford, M. and Wyeth, G. (2008). Mapping a suburb with a single camera using a biologically inspired SLAM system. *IEEE Transactions on Robotics*, 24(5):1038–1053.

Murillo, A. C., Sagüés, C., Guerrero, J. J., Goedemé, T., Tuytelaars, T., and Van Gool, L. (2007). From omnidirectional images to hierarchical localization. *Robotics and Autonomous Systems*, 55(5):372–382.

Murphy, K., Torralba, A., Eaton, D., and Freeman, W. (2006). Object detection and localization using local and global features. In *CLOR06*, pages 382–400.

Nayar, S., Nene, S., and Murase, H. (1995). Subspace methods for robot vision. Technical Report CUCS-06-95, Department of Computer Science, Columbia University.

Neira, J. and Tardós, J. (2001). Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, Vol. 17(No. 6):pp. 890 – 897.

Newman, P., Cole, D., and Ho, K. (2006). Outdoor SLAM using visual appearance and laser ranging. In *ICRA*, pages 1180–1187, Orlando, Florida, USA. IEEE.

Newman, P. and Ho, K. (2005). SLAM - Loop closing with visually salient features. In *ICRA*, Barcelona, Spain. IEEE.

Newman, P. M. (1999). *On the structure and solution of the simultaneous localization and mapping problem.* PhD thesis, University of Sydney, Australia.

Ni, K., Steedly, D., and Dellaert, F. (2007). Out-of-core bundle adjustment for large-scale 3d reconstruction. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1–8, Rio de Janeiro, Brazil. IEEE.

Nistér, D. (2004). An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(6):756–777.

Nistér, D., Naroditsky, O., and Bergen, J. R. (2004). Visual odometry. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–659, Washington, DC, USA. IEEE.

Olson, E. B. (2008). Implicit data association from spectrally clustered local matches. In *Inside Data Association Workshop: Robotics Science and Systems*.

Olson, E. B. (2009a). Real-time correlative scan matching. In *ICRA*, Kobe, Japan. IEEE.

Olson, E. B. (2009b). Recognizing places using spectrally clustered local matches. *Robotics and Autonomous Systems*, 57(12):1157 – 1172. Inside Data Association.

Ortín, D. and Montiel, J. M. M. (2001). Indoor robot motion based on monocular images. *Robotica*, 19(3):331–342.

Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. (2007). Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, Minnesota, USA. IEEE Computer Society.

Pollefeys, M., Nistér, D., Frahm, J. M., Akbarzadeh, A., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Kim, S. J., Merrell, P., Salmi, C., Sinha, S., Talton, B., Wang, L., Yang, Q., Stewénius, H., Yang, R., Welch, G., and Towles, H. (2008). Detailed real-time urban 3d reconstruction from video. *Int. J. Comput. Vision*, 78(2-3):143–167.

Pronobis, A., Caputo, B., Jensfelt, P., and Christensen, H. I. (2010). A realistic benchmark for visual indoor place recognition. *Robotics and Autonomous Systems (RAS)*, 58(1):81–96.

Ramisa, A., Tapus, A., Aldavert, D., Toledo, R., and de Mántaras, R. L. (2009). Robust vision-based localization using combinations of local feature regions detectors. *Autonomous Robots Journal*, 27:373–385.

Ranganathan, A. and Dellaert, F. (2007). Semantic modeling of places using objects. In *Proceedings of Robotics: Science and Systems (RSS)*, Atlanta, USA.

Rogers, J. G. and Christensen, H. I. (2009). Normalized graph cuts for visual slam. In *IROS*, pages 918–923, St. Louis, USA. IEEE/RSJ.

Russell, B. C., Torralba, A. B., Murphy, K. P., and Freeman, W. T. (2008). Labelme: A database and web-based tool for image annotation. *International Journal of Computer Vision*, 77(1-3):157–173.

Saffiotti, A. and LeBlanc, K. (2000). Active perceptual anchoring of robot behavior in a dynamic environment. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3796–3802, San Francisco, CA. Online at http://www.aass.oru.se/˜asaffio/.

Salvi, J., Petillot, Y. R., and Batlle, E. (2008). Visual SLAM for 3D large-scale seabed acquisition employing underwater vehicles. In *IROS*, pages 1011–1016, Nice, France. IEEE/RSJ.

Scaramuzza, D., Fraundorfer, F., and Pollefeys, M. (2010). Loop in appearance-guided omnidirectional visual odometry by using vocabulary trees. *Robotics and Autonomous System*. To appear.

Scaramuzza, D., Siegwart, R., and Martinelli, A. (2009). A robust descriptor for tracking vertical lines in omnidirectional images and its use in mobile robotics. *Int. J. Rob. Res.*, 28(2):149–171.

Schaffalitzky, F. and Zisserman, A. (2002). Multi-view matching for unordered image sets, or "How do I organize my holiday snaps?". In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 1, pages 414–431, Copenhagen, Denmark. Springer.

Schindler, G., Brown, M., and Szeliski, R. (2007). City-scale location recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, Minnesota, USA. IEEE Computer Society.

Se, S., Lowe, D., and Little, J. (2002). Global localization using distinctive visual features. In *IROS*, pages 226–231, Lausanne, Switzerland. IEEE/RSJ.

Segvic, S., Remazeilles, A., Diosi, A., and Chaumette, F. (2009). A mapping and localization framework for scalable appearance-based navigation. *Computer Vision and Image Understanding*, 113(2):172–187.

Shi, J. and Tomasi, C. (1994). Good features to track. In *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 593 – 600.

Sim, R. and Dudek, G. (1999). Learning and evaluating visual features for pose estimation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1217–1222, Kerkyra, Greece. IEEE.

Sim, R., Griffin, M., Shyr, A., and Little, J. J. (2005). Scalable real-time vision-based SLAM for planetary rovers. In *IEEE IROS Workshop on Robot Vision for Space Applications*, pages 16–21, Edmonton, AB. IEEE, IEEE Press.

Sivic, J. and Zisserman, A. (2003). Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision (ICCV)*, volume 2, pages 1470–1477, Nice, France. IEEE.

Smeulders, A. W. M., Worring, M., Santini, S., Gupta, A., and Jain, R. (2000). Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(12):1349–1380.

Smith, R., Self, M., and Cheeseman, P. (1990). *Estimating uncertain spatial relationships in robotics*, pages 167–193. Springer-Verlag New York, Inc., New York, NY, USA.

Snavely, N., Seitz, S. M., and Szeliski, R. (2008). Skeletal sets for efficient structure from motion. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, Anchorage, Alaska, USA. IEEE.

Spexard, T., Li, S., Wrede, B., Fritsch, J., Sagerer, G., Booij, O., Zivkovic, Z., Terwijn, B., and Kröse, B. (2006). Biron, where are you? - enabling a robot to learn new places in a real home environment by integrating spoken dialog and visual localization. In *IROS*, Beijing, China. IEEE/RSJ.

Stachniss, C., Martínez-Mozos, O., Rottmann, A., and Burgard, W. (2005). Semantic labeling of places. In *Proceedings of the International Symposium on Robotics Research*, San Francisco, CA, USA.

Stewénius, D. H., Engels, C., and Nistér, D. D. (2006). Recent developments on direct relative orientation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60(4):284–294.

Sturm, J. and Visser, A. (2009). An appearance-based visual compass for mobile robots. *Robot. Auton. Syst.*, 57(5):536–545.

Tapus, A. and Siegwart, R. (2006). A cognitive modeling of space using fingerprints of places for mobile robot navigation. In *ICRA*, Orlando, Florida, USA. IEEE.

Thrun, S. (2002). Robotic mapping: A survey. In Lakemeyer, G. and Nebel, B., editors, *Exploring Artificial Intelligence in the New Millenium*, pages 1–35. Morgan Kaufmann.

Topp, E. and Christensen, H. (2010). Detecting region transitions for human-augmented mapping. *IEEE Transactions on Robotics*, 26(4):715–720.

Topp, E., Huettenrauch, H., Christensen, H., and Eklundh, K. S. (2006). Bringing together human and robotic environment representations - a pilot study. In *In Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Torr, P. H. S. and Murray, D. W. (1997). The development and comparison of robust methods for estimating the fundamental matrix. *Int. Journal of Computer Vision*, 24(3):271–300.

Torralba, A., Murphy, K., Freeman, W., and Rubin, M. (2003). Context-based vision system for place and object recognition. In *Proceedings of the International Conference on Computer Vision (ICCV)*, Nice, France. IEEE.

Triggs, B., McLauchlan, P. F., Hartley, R. I., and Fitzgibbon, A. W. (2000). Bundle adjustment - a modern synthesis. In *ICCV '99: Proceedings of the International Workshop on Vision Algorithms*, pages 298–372, London, UK. Springer-Verlag.

Ulrich, I. and Nourbakhsh, I. (2000). Appearance-based place recognition for topological localization. In *ICRA*, volume 2, pages 1023 – 1029, San Francisco, California, USA.

Valgren, C., Duckett, T., and Lilienthal, A. (2007). Incremental spectral clustering and its application to topological mapping. In *ICRA*, pages 4283–4288, Roma, Italy. IEEE.

Valgren, C. and Lilienthal, A. J. (2007). SIFT, SURF and Seasons: Long-term outdoor localization using local features. In *Proceedings of the 3rd European Conference on Mobile Robots*, pages 253–258.

Valgren, C. and Lilienthal, A. J. (2008). Incremental spectral clustering and seasons: Appearance-based localization in outdoor environments. In *ICRA*, pages 1856–1861, Pasadena, California, USA. IEEE.

Valgren, C. and Lilienthal, A. J. (2010). SIFT, SURF & Seasons: Appearance-based long-term localization in outdoor environments. *Robotics and Autonomous Systems*, 58(2):157–165.

Vasudevan, S., Gachter, S., Berger, M., and Siegwart, R. (2006). Cognitive maps for mobile robots - an object based approach. In *Proceedings of the IEEE IROS 2006 workshop - From Sensors to Human Spatial Concepts (FS2HSC 2006)*, Beijing, China.

Vasudevan, S. and Siegwart, R. (2008). Bayesian space conceptualization and place classification for semantic maps in mobile robotics. *Robotics and Autonomous Systems*, 56(6):522–537.

Verbeek, J. (2004). *Mixture models for clustering and dimension reduction*. PhD thesis, University of Amsterdam.

Yairi, T. (2007). Map building without localization by dimensionality reduction techniques. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 1071–1078, New York, NY, USA. ACM.

Zender, H., Mozos, O. M., Jensfelt, P., Kruijff, G.-J. M., and Burgard, W. (2008). Conceptual spatial representations for indoor mobile robots. *Robotics and Autonomous Systems*, 56(6):493–502.

Zhang, Z. (1998). Determining the epipolar geometry and its uncertainty: A review. *Int. J. Comput. Vision*, 27(2):161–195.

Zheng, Y., M.ao, Y.ng, and Adam, H. (2009). Tour the world: building a web-scale landmark recognition engine. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami, USA. IEEE.

Zhou, W., Miro, J., and Dissanayake, G. (2008). Information-efficient 3-d visual SLAM for unstructured domains. *IEEE Transactions on Robotics*, 24(5):1078–1087.

Zitova, B. and Flusser, J. (2003). Image registration methods: a survey. *Image and Vision Computing*, 21(11):977–1000.

Zivkovic, Z., Bakker, B., and Kröse, B. (2005). Hierarchical map building using visual landmarks and geometric constraints. In *IROS*, Edmonton, Canada. IEEE/RSJ.

Zivkovic, Z., Bakker, B., and Kröse, B. (2006). Hierarchical map building and planning based on graph partitioning. In *ICRA*, pages 803–809, Orlando, Florida, USA. IEEE.

Zivkovic, Z. and Booij, O. (2005). How did we built our hyperbolic mirror omnidirectional camera - practical issues and basic geometry. Technical Report IAS-UVA-05-04, University of Amsterdam.

Zivkovic, Z., Booij, O., Kröse, B., E.Topp, and H.I.Christensen (2008). From sensors to human spatial concepts: an annotated dataset. *IEEE Transactions on Robotics*, 24(2):501–505.

Summary

A common task for a mobile robot, is to build a map of its environment. It needs such a map to localize itself in the environment, and to navigate to a specific place. In this thesis, we propose methods to build such a map on the basis of images. These images are taken with a camera that is mounted onto the robot. The methods are specifically developed for robots operating in real home environments, such as autonomous vacuum cleaners. Therefore, all methods are evaluated on image sets which were acquired in real home environments.

Building a map is particularly difficult in human inhabited home environments. We can identify two main challenges. First, the captured images are often of poor quality. As opposed to brightly lit office spaces and outdoor environments, home environments usually have bad lighting condition. On top of this, humans can block the view of the robot and change the appearance of the environment, for example by switching off lights. Such circumstances call for a robust mapping approach. Second, the robot should be able to communicate with humans about the map and add labels, such as "kitchen" or "chair", to the map. To allow for such communication, during map building, the method should be efficient enough to perform in real-time.

Different types of maps have been proposed to solve this robot task. In this thesis, we focus on a specific type of map termed the "view-based topological map". In Chapter 2, we see that this is a commonly used map in the field of vision-based robotics. It can be seen as a graph for which each node denotes an image and each link between two images denotes that they depict an overlapping part of the environment. The basic task of map-building and localization is finding for a new image all other images that could link with it. This later task is called "view-based data association" and depends on the ability to compare images with each other.

Two images can be compared by automatically extracting a set of salient image points from both of them and matching these sets to obtain a number of image point correspondences. If two images depict an overlapping part of the environment, then the spatial layout of the correspondences depends greatly on the relative robot pose. This relative pose can be parameterized by a rotation and a translation up to scale. To make image comparison robust and efficient, it is beneficial to incorporate constraints on the possible robot poses. We thoroughly investigate how the assumption that the robot moves over a planar surface can be used to improve existing algorithms. We derive an algorithm that computes the planar relative pose given only two correspondences and combine it with the well-known RANSAC algorithm (RANdom SAmple Consensus). On top of that, we propose an efficient method based on the Hough Transform that determines a probability density over the space of all planar relative poses. On the one hand, this can be used to estimate the relative pose given two images. In Chapter 3, we show that our relative pose solution is more accurate and efficient than the popular RANSAC-based method. On the other hand, it can be used as an image similarity measure, by estimating the probability that robot pose is indeed correct. In Chapter 4, we show that this image similarity measure is better than existing measures used for both topological view-based mapping and the related task of image-based place recognition.

Even if an efficient image comparison method is used, for very large view-based maps with thousands of images, it becomes infeasible to compare with all images. In Chapter 5, we propose a method that finds a set of key images that best represents the complete set of images of the map. This set of key images defines a subgraph of the complete graph representing the map. It is based on a graph theoretic technique called the "Connected Dominating Set" (CDS). We explain why this algorithm is the optimal choice and how it can be used for view-based mapping and localization procedures. The results of these procedures are close to the ones obtained when comparing with all images, while being an order of magnitude faster.

Combining the image comparison technique with the CDS method, results in scalable real-time data association. We show, in two appendices, that this can be used to perform real-time interactive map-building and goal-directed navigation. In addition, the proposed data association method could be used to improve existing SLAM systems (Simultaneous Localization And Mapping). This would allow for efficient and robust 3D geometric map-building of real home environments.

Samenvatting[2]

Een standaard taak voor een mobiele robot is het maken van een kaart van zijn omgeving. Deze heeft de robot nodig om zich te kunnen lokaliseren in zijn omgeving en om te kunnen navigeren naar een bepaalde plek. Dit proefschrift presenteert methodes voor het maken van zo'n kaart op basis van foto's. Deze foto's worden gemaakt met behulp van een camera die op de robot zelf is gemonteerd. De methodes zijn specifiek bedoeld voor robots die bij mensen thuis rondrijden, zoals automatische stofzuigers. Voor het testen van de methodes wordt daarom gebruik gemaakt van foto's die in echte huizen zijn genomen.

Een kaart maken van foto's van een bewoond huis is moeilijk. Er zijn twee problemen. Ten eerste zijn foto's die de robot in een huis maakt vaak van een relatief slechte kwaliteit. In vergelijking met goed verlichte kantoorruimtes en buiten omgevingen, zijn huizen meestal slecht verlicht. Bovendien kunnen mensen het zicht van de robot ontnemen en ook de omgeving zelf veranderen, bijvoorbeeld door lichten uit te doen. Zulke omstandigheden vragen om een robuuste aanpak voor het maken van kaarten. Ten tweede moet de robot kunnen communiceren met mensen over de kaart en labels kunnen toevoegen, zoals "keuken" of "stoel". Om er voor te zorgen dat zulke communicatie al tijdens het kaart maken kan plaats vinden, moet de aanpak efficiënt genoeg zijn om in real-time te kunnen draaien.

Standaard robot taken, zoals lokalisatie, kunnen met behulp van verschillende soorten kaarten worden uitgevoerd. In dit proefschrift gebruiken we een specifiek soort kaart, genaamd "view-based topological map". In Hoofdstuk 2 laten we zien dat dit een veel gebruikt type kaart is in het onderzoeksveld van de robotica. Het is eigenlijk een soort graaf waarin de knopen de foto's voorstellen en een verbinding tussen twee knopen aangeeft dat de foto's een overlappend deel van de omgeving tonen. De kerntaak voor het maken van zo'n kaart is om voor elke nieuw genomen foto, bestaande foto's te vinden, waarmee het verbonden kan worden. Dit laatste wordt "view-based data association" genoemd en is afhankelijk van de mogelijkheid om twee foto's met elkaar te vergelijken.

Twee foto's kunnen met elkaar vergeleken worden door kenmerkende beeldpunten van beide foto's te vergelijken om zodoende een set van paren beeldpunten te krijgen. Als de foto's een overlappend deel van de omgeving laten zien, dan zullen de posities van deze paren beeldpunten afhankelijk zijn van de relatieve pose van de robot. Deze relatieve pose kan beschreven worden door een relatieve rotatie en een translatie van onbepaalde grootte. Om de vergelijking van foto's robuust en efficiënt te maken, is het handig om het aantal mogelijke robot poses te beperken. Wij hebben grondig onderzocht hoe we gebruik kunnen maken van het feit dat de robot over een vlakke vloer rijdt en dus alleen planaire bewegingen kan maken. We hebben een algoritme afgeleid dat de planaire relatieve pose kan bepalen uit twee punt-paren en hebben dit gecombineerd met het bekende RANSAC algoritme (RANdom SAmple Consensus). Bovendien

[2]In dutch

hebben we een efficiënte methode ontwikkeld op basis van de Hough Transform, om de kansdichtheid te bepalen over alle mogelijke planaire relatieve poses. Aan de ene kant kan deze methode gebruikt worden voor het schatten van de juiste relatieve pose. In Hoofdstuk 3 laten we zien dat onze oplossing hierin nauwkeuriger en efficiënter is dan de populaire op RANSAC gebaseerde methode. Aan de andere kan kan deze methode ook gebruikt worden om foto's te vergelijken. Dit wordt gedaan op basis van de kans van de geschatte pose. In Hoofdstuk 4 laten we zien dat deze vergelijkingsmaat beter werkt dan bestaande maten, die gebruikt worden voor "view-based mapping", en het daaraan verwante "image-based place recognition".

Maar zelfs met deze efficiënte methode, is het zo dat bij grote kaarten bestaande uit duizenden foto's, het niet praktisch is om alle foto's met elkaar te vergelijken. In Hoofdstuk 5 presenteren we een methode dat een selectie van de foto's maakt die het beste de volledig set van foto's beschrijft. Deze subset van foto's definieert een subgraaf van de complete graaf van de "view-based map". Deze subgraaf wordt gevonden door middel van een techniek uit de graven-theorie, genaamd "Connected Dominating Set" (CDS). We leggen uit waarom deze methode de optimale oplossing produceert en hoe het gebruikt kan worden in procedures om een kaart te maken en om te kunnen lokaliseren. De uitkomsten van die procedures wijken nauwelijks af van die verkregen met een naïve aanpak, waarin alle foto's met elkaar worden vergeleken. Echter onze procedures zijn een orde van grootte sneller.

Door de ontwikkelde methodes te combineren ontstaat een schaalbare real-time oplossing voor de "data association" taak. In twee appendices tonen we aan dat zulke oplossingen gebruikt kunnen worden om op een interactieve manier een kaart van een huis te maken en doelgericht te kunnen navigeren. Bovendien zijn de methodes zeer bruikbaar voor het verbeteren van bestaande SLAM applicaties (Simultaneous Localization And Mapping). Dit kan resulteren in een efficiënte en robuuste manier voor het maken van geometrische modellen in 3D van bewoonde huizen.