MULTI-CAMERA SIMULTANEOUS LOCALIZATION AND MAPPING

Brian Sanderson Clipp

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2010

Approved by:

Marc Pollefeys

Jan-Michael Frahm

Gary Bishop

Svetlana Lazebnik

Jongwoo Lim

Gregory Welch

ABSTRACT

BRIAN SANDERSON CLIPP: Multi-Camera Simultaneous Localization and Mapping
(Under the direction of Marc Pollefeys and Jan-Michael Frahm)

In this thesis, we study two aspects of simultaneous localization and mapping (SLAM) for multi-camera systems: minimal solution methods for the scaled motion of non-overlapping and partially overlapping two camera systems and enabling online, real-time mapping of large areas using the parallelism inherent in the visual simultaneous localization and mapping (VSLAM) problem.

We present the only existing minimal solution method for six degree of freedom structure and motion estimation using a non-overlapping, rigid two camera system with known intrinsic and extrinsic calibration. One example application of our method is the three-dimensional reconstruction of urban scenes from video. Because our method does not require the cameras' fields-of-view to overlap, we are able to maximize coverage of the scene and avoid processing redundant, overlapping imagery.

Additionally, we developed a minimal solution method for partially overlapping stereo camera systems to overcome degeneracies inherent to non-overlapping two-camera systems but still have a wide total field of view. The method takes two stereo images as its input. It uses one feature visible in all four views and three features visible across two temporal view pairs to constrain the system camera's motion. We show in synthetic experiments that our method creates rotation and translation estimates that are more accurate than the perspective three-point method as the overlap in the stereo camera's fields-of-view is reduced.

A final part of this thesis is the development of an online, real-time visual SLAM system that achieves real-time speed by exploiting the parallelism inherent in the VSLAM

problem. We show that feature tracking, relative pose estimation, and global mapping operations such as loop detection and loop correction can be effectively parallelized. Additionally, we demonstrate that a combination of short baseline, differentially tracked corner features, which can be tracked at high frame rates and wide baseline matchable but slower to compute features such as the scale-invariant feature transform (SIFT) (Lowe, 2004) can facilitate high speed visual odometry and at the same time support location recognition for loop detection and global geometric error correction.

To Rachel.

ACKNOWLEDGEMENTS

I would like to thank my advisor Marc Pollefeys and co-advisor Jan-Michael Frahm for their support of this work. There were times, particularly close to conference deadlines, that it seemed like some of the methods developed in this dissertation might not succeed. I am particularly grateful for Marc and Jan's suggestions and encouragement at these times that helped me push through to solutions.

Thanks also to my committee members, Gary Bishop, Jongwoo Lim, Lana Lazebnik and Gregory Welch, who gave helpful advice and suggestions for this work.

My co-authors have been instrumental in completing this dissertation. This list of co-authors includes Richard Hartley, Jae-Hak Kim, Jongwoo Lim, Jan-Michael Frahm, Marc Pollefeys, Rahul Raguram, Gregory Welch, and Christopher Zach all of whom I would like to thank. Thanks in particular to Christopher Zach who helped me turn my geometric intuition into working algebraic solution methods. Also, thanks to Greg for the many interesting discussions we have had over the last few years. I am also grateful to Phillipos Mordohai, who as a post doc at UNC, along with Marc and Jan-Michael, helped me to gain my footing in multi-view geometry and computer vision.

My work in multi-camera systems involved building a lot of strange contraptions including a backpack mounted data collection system, a helmet mounted stereo camera, and a roof top recording platform for use on the department van. I owe a great deal of gratitude to John Thomas who built these systems that helped to make many of my experiments, whether presented in this dissertation or not, possible. Thanks John.

David Gallup and I have shared an office since July of 2005 when I came to UNC. Thanks Dave for being a good friend, putting up with whatever annoying habits I am sure

I have, and listening to my sometimes repetitive stories. I am going to miss our occasional procrastination sessions and mental breaks disguised as philosophical discussions.

Additionally, I would like to thank my family for their constant support and encouragement as I pursued my graduate studies. You all helped to keep me sane through this process by reminding me there are important things outside of academia.

Finally, to my wife Rachel, I could not have done this without your patience, sacrifice, understanding, and support. I am so glad you were on board with going to graduate school together. Sharing this experience has been wonderful.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CCD Charge-Coupled Device

CUDA Compute Unified Device Architecture

DOF Degree of Freedom

DoG Difference of Gaussian

EKF Extended Kalman Filter

FAB-MAP Fast Appearance Based Mapping

fps Frames per Second

GNC Graduated Non-Convexity

GPU Graphics Processing Unit

GS Global SLAM

ICP Iterative Closest Point

KLT Kanade, Lukas, Tomasi feature tracker

LIDAR Light Direction and Ranging

MSER Maximally Stable Extremal Region

PTAM Parallel Tracking and Mapping

RANSAC Random Sample Consensus

SBA Sparse Bundle Adjustment

SF Scene Flow

SfM Structure from Motion

SIFT Scale-Invariant Feature Transform

SLAM Simultaneous Localization and Mapping

TF-IDF Term-Frequency Inverse Document Frequency

VIP Viewpoint Invariant Patch

VO Visual Odometry

VSLAM Visual Simultaneous Localization and Mapping

CHAPTER 1

Introduction

Visual Simultaneous Localization and Mapping (VSLAM) is the problem of using a moving sensor system with one or more cameras to map an unknown environment and simultaneously keep track of the sensor system's pose within the map. The sensor system might be as simple as a single camera or could be a multi-camera system including other sensors such as accelerometers, gyroscopes, and wheel encoders. Like many problems in artificial intelligence VSLAM is something that most humans do fairly easily but is highly complex and difficult to automate.

The more general simultaneous localization and mapping (SLAM) problem has been studied extensively in the robotics community (Kaess et al., 2007; Paskin, 2003; Thrun et al., 2005; Smith and Cheeseman, 1987). The sensors used in SLAM typically include Light Direction and Ranging (LIDAR), acoustic range sensors, bump sensors, as well as accelerometers, gyroscopes and wheel encoders. What sets apart visual SLAM is the use of cameras as sensors. In contrast to LIDAR, cameras are purely passive sensors and so do not emit any electromagnetic radiation. Because cameras are non-emissive, they typically require less power and are suitable for applications where stealth or a lack of interference between multiple systems is crucial. Additionally, cameras are less expensive than specialized LIDAR sensors and are more pervasive in our world today. Most people today carry a mobile phone that includes a camera, which can be used for SLAM, as well as location recognition, which can support location-based services.

The peculiarities of cameras, in comparison to other sensing modalities, make the

VSLAM problem a separate class of problem from general SLAM. Cameras provide bearing-only information, e.g. the direction to a target but not the distance to the target. Cameras also have effectively unlimited range; they detect the first object a ray encounters as it emanates from the camera. In contrast, the range of LIDAR and acoustic sensors is limited by the amount of energy the sensor can broadcast into the environment and the reflectivity or absorbtion of the environment's surfaces. This limited range actually simplifies the SLAM problem since only what is near the sensor can be measured by the sensor. This can lead to certain subdivisions of the map, which can simplify the SLAM problem. In contrast, the position of a camera system may have little to do with the spatial distribution of the objects it measures.

The VSLAM problem is important because it has applications in augmented reality, robotic navigation, remote sensing, and generating dense three-dimensional models from video. In augmented reality, a user views the world through some form of output device, generally either with a head-mounted display or hand-held device such as a cell phone. Synthetic objects are then placed on top of the real-scene in the user's view. These objects could include information about the environment or synthetic game characters. In any case, to insert synthetic objects accurately, SLAM must be used to measure the pose of the display device in the environment. Visual SLAM (VSLAM) is an attractive option for augmented reality because of the low cost and power requirements of cameras and their relatively high angular resolution.

SLAM is also necessary for a robotic system to autonomously navigate its environment. It must have some way to create a map of its surroundings and measure its pose in the environment. The use of cameras in SLAM for robots is motivated by many of the same factors as in augmented reality. In particular, low power requirements can drive the choice of using VSLAM.

The VSLAM problem is known in the vision community as Structure from Motion (SfM) and is the first step toward creating three-dimensional models of the world from

Figure 1.1: Textured 3D models reconstructed from multiple images.

video. Given the camera poses from VSLAM dense image matching can be performed to find the depth of the scene with respect to the cameras, and given the camera poses the shape of the scene can be recovered in a global coordinate frame. Once the scene shape is recovered, it can be textured with the imagery to create visually appealing virtual models of the measured environment. Some example models are shown in Figure 1.1.

This thesis introduces the VSLAM problem and addresses two fundamental issues in VSLAM. The first is the trade-off in two camera systems between field-of-view overlap and accurate scaled motion estimation. We show this trade-off to be false and that non-overlapping and partially overlapping two-camera systems can be used in absolutely scaled VSLAM. The second issue is real-time performance. Through a principled analysis of the VSLAM problem, we show how a combination of tolerable latency, parallelism, and integration of 3D pose estimation with 2D feature matching can accelerate six degree of freedom (DOF) VSLAM to a previously unachieved level of performance combining speed with accurate structure and motion computation.

CHAPTER 2

The Visual SLAM Problem

The "Visual SLAM" problem, which is also known as "Structure from Motion", has been studied extensively in the robotics and computer vision fields. This chapter will give a brief history of the VSLAM problem as well as introduce the state of the art in VSLAM. It will then discuss the structure of the VSLAM problem and the various sub-processes that must be done in any VSLAM system namely, correspondence finding, relative pose estimation, and global mapping.

Harris and Pike demonstrated one of the first VSLAM methods on an image sequence (Harris and Pike, 1988). Their work contained many of the major components of a VSLAM system including feature matching, relative pose estimation, and a Kalman filter based method for fusing the measurements from multiple views. Using a stereo camera, their system created a map of point and line features with covariance matrices representing their uncertainties. However, they neglected the correlations between features which can create problems.

With estimated correlations between features, if feature A is detected in an image but not feature B, then the measurement update of A can be propagated to an update of B through their covariance. This reflects what we would expect. If the system has previously build a map of the outside of my home and it detects my home's front door in an image, then that also gives information about where the front window is even if the window was not seen in the same image as the door. Without modeling the correlations between window and door we may become over-confident in the door's position with respect to the window

and when we finally see the window reject it's features as outliers.

Another area where correlations are critical is in loop completion. Consider the simple case were a camera is panned around the vertical so that it views the walls of a room. As it turns it builds a map of the walls and its pose. The farther it turns away from the origin (its starting pose) the more uncertain its pose is as well as its feature estimates. When the camera turns all the way around and re-detects features that were mapped in the first image this completes a loop. Recognizing that the features at the end of the loop are the same as those at the beginning should reduce the uncertainty of the features seen in the frames just before the loop was completed as well as update their positions. Without modeling the correlations between features this update is impossible and the loop cannot be properly completed. As stated by Davison (2007), ignoring the correlations between features could lead to over-confidence in the feature estimates' accuracy and the inability to close loops or detect drift.

Azarbayejani and Pentland presented an Extended Kalman Filter (EKF) approach to recursively estimate the structure, camera motion and camera focal length from an image sequence (Azarbayejani and Pentland, 1995). Davison (2007) was the first to demonstrate a real-time VSLAM system using a single camera as its only measurement device. His work was also based on an Extended Kalman Filter and could map areas the size of a desktop or small room, detecting and completing loops. Davison's system modeled the correlations between mapped features. This made the filter's estimate of uncertainty consistent, but it limited the number of features that could be mapped in real time to less than one-hundred.

A particle filter approach to VSLAM was presented by Eade and Drummond (2006). Their system could also map small office scale environments but the small number of particles that could be processed in real time limited their map size.

Recent work on VSLAM has focused primarily on overcoming speed limitations as the number of cameras or 3D features in the map increases. Clemente et al. (Clemente et al., 2007) proposed a sub-map approach where the total map is made up of a set of smaller

Euclidean maps connected by transformations. Since each Euclidean map is limited in size it can be updated in real-time. A major drawback of this approach is that global correction is done by fixing the sub-maps and varying the transformations between them. This forces all of the error accumulated in each of the sub-maps into the joints between the maps.

Eade and Drummond (2007) used a sub-map based approach to accelerate the mapping process in. In their work, each sub-map contained about fifty 3D features with their associated uncertainties and correlations stored in a covariance matrix. They used the Laplacian of the projection function as a measure of the non-linearity of the projection function in order to decide where to split the map into sub-maps. By limiting the sub-map sizes they can fold the information from a new image into the map in real-time. Additionally, they can optimize the sub-maps with respect to each other to arrive at a globally consistent map. This is critical because inconsistency can lead to problems in detecting and correcting loops.

Klein and Murray (2007) developed a system for VSLAM they call "Parallel Tracking and Mapping" or PTAM. In PTAM, online, real-time camera pose estimation (tracking) and structure estimation (mapping) are separated into two threads. During tracking, the camera's pose is estimated by matching features extracted in the current image to features in the map. The map is represented as a set of point features and key-frame camera poses. Batch optimization processes such as bundle adjustment can be used to optimize the mapped feature positions and key-frame poses in the images in a second parallel thread. Bundle adjustment is a process that minimizes the difference between the measured and expected projections of the 3D point features by varying the camera poses and 3D feature locations in the map. This optimization need not be real-time since the camera tracking is done in real-time in the first thread. Klein and Murray developed PTAM for use in augmented reality in small workspaces. Their use of parallel threads allows the camera pose estimation to run at frame rate to support adding synthetic objects to the scene while also refining the map using consistent, batch optimization. Klein and Murray (2009) developed a version of PTAM for the iPhone which shows great potential as an augmented reality platform.

The VSLAM method presented in Chapter 5 differs from PTAM in that we can explore new areas while the map is globally corrected. In PTAM new key-frames can only be added between global map corrections, limiting the rate of key-frame addition as the map grows. In our method global map correction must complete between new loop completions rather than between new key-frames, significantly increasing the size of areas that can be mapped online and in real-time.

Scalability of performance to larger map sizes was a focus of Konolige and Agrawal's method called "FrameSLAM" (2008). Like Klein and Murray, they also used selected key-frames from the image sequence and only included these in the map. Konolige and Agrawal's innovation was to convert the constraints imposed by corresponding feature measurements in two images into a synthetic measurement with an associated uncertainty tying the two camera poses together with a sort of virtual spring. This permanently marginalizes out the features from the map, significantly speeding up the minimization used to correct for loops in the camera's path. Their method achieves this speedup at the cost of a less accurate map structure. Accuracy is reduced because the feature measurements cannot be re-linearized once the features are permanently marginalized out of the optimization.

2.1 Basic Structure of the VSLAM Problem

Visual SLAM can be broken up into three primary operations: correspondence finding, relative pose estimation, and global mapping. Correspondence finding involves determining which areas of the current image correspond to known features in the map or to areas in the previous image. Relative pose estimation uses these correspondences to find the camera motion from one frame to the next. Global mapping involves detecting long-range loops in the camera's path and correcting the map to reflect these loops and eliminate long-term mapping error. Figure 2.1 shows the basic architecture of a VSLAM system and the data that passes between the modules. Sections 2.2 through 2.5 will introduce these operations

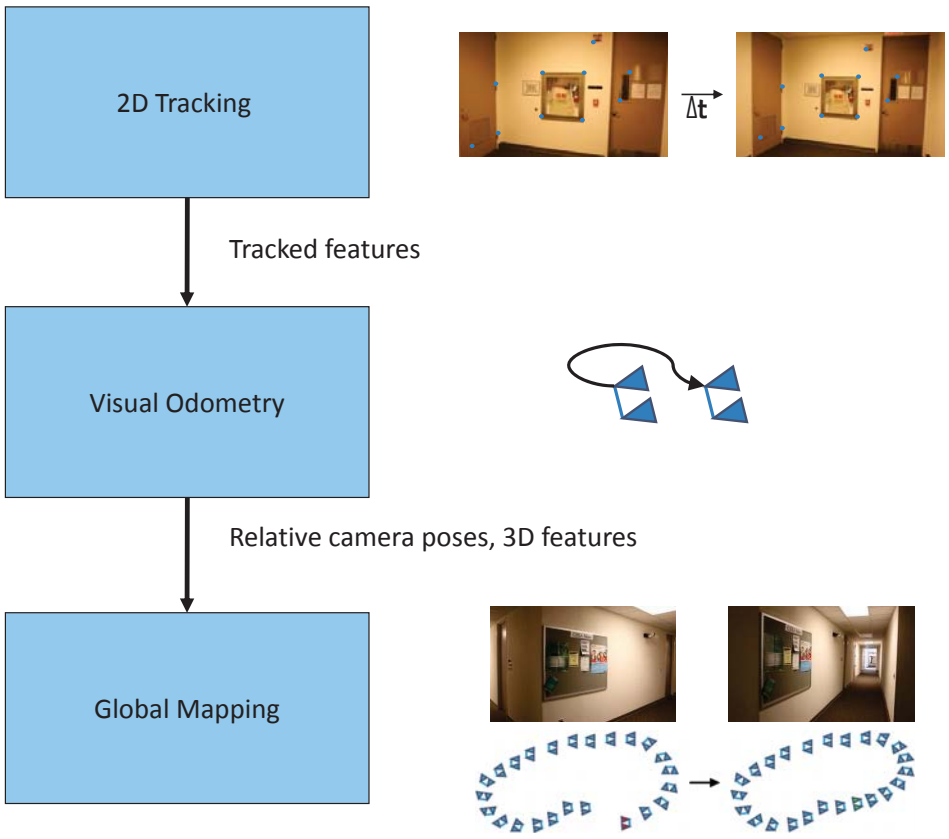

Figure 2.1: Basic VSLAM operations and data that passes between them

and discuss relevant prior work.

Any visual SLAM approach must complete each of the three above operations. Some such as the one we have developed (Clipp et al., 2010) treat each of the operations as largely independent with well-defined but limited data transfer between operations. Others, such as the method of Davison et al. (2007) perform the correspondence finding, relative pose, and global mapping operations in a single, unified manner. Both the separate and unified approaches have benefits. Separating correspondence finding, relative pose estimation and global mapping operations introduces parallelism to VSLAM that can improve processing speed as well as allow for exploration while the map is globally updated after a loop is completed. However, a unified approach also has benefits. For example, tying pose prediction to correspondence finding can aid in tracking features when camera motion is erratic.

2.2 Correspondence Finding

There are two major types of correspondence finding approaches: unguided approaches that find correspondences without modeling inter-correspondence relationships, and those guided by and coupled to 3D relative camera pose estimation. Unguided approaches were the first to be developed by the computer vision community so they will be described first.

2.2.1 Unguided Correspondence

Unguided correspondence methods estimate optical flow without any camera motion model or other external sensors, e.g. inertial sensors. Unguided approaches can be roughly divided into two classes: differential-tracking approaches that depend on a small feature motion assumption, and matching approaches that can find correspondence in pairs of images with larger viewpoint changes.

In Kanade, Lukas, Tomasi (KLT) feature tracking (Lucas and Kanade, 1981; Tomasi and Kanade, 1991), a sparse set of corner features are extracted from the image. These features have a strong gradient in two orthogonal directions and so are well constrained in the image (Shi and Tomasi, 1994). Newton's method is then used to find the minimum of the cost function $\epsilon = \int \int_W [J\left(x + \frac{d}{2}\right) - I\left(x - \frac{d}{2}\right)]^2 w\left(x\right) dx$ and solve for the feature displacement from one image to the next in a temporal sequence. This equation integrates the difference between patches of images I and J over window size W parameterized by their center location x and a displacement d. The KLT tracking formulation assumes sub-pixel feature from image to image. A scale space pyramid is used for tracking motions greater than a single pixel. The KLT feature tracker treats each feature in an image independently and so can be easily parallelized and mapped to the graphics processor to achieve fast tracking. One implementation achieves greater than two-hundred frames per second tracking, including estimating the mean intensity change for all features on monocular 720x560 pixel resolution video (Zach, 2009).

Many approaches exist for finding feature correspondences over wide baselines. Harris features (Harris and Stephens, 1988) are extracted at strong corners in the image as measured by the gradient magnitude in a local region. Harris corners use an approximation to the eigenvalues of the structure tensor to detect features. For this reason, they give a less accurate response than Tomasi and Shi's operator (1994) but are more computationally efficient.

One of the limitations of Harris features is that they are point features, lacking an intrinsic scale or neighborhood size. Lowe (2004) showed how a scale space approach to detecting blob-like features using convolution with a difference of Gaussian (DoG) operator could lead to scale invariance. The difference of Gaussian operator is an approximation to the Laplacian of Gaussian operator and so reacts strongly to blobs. The equation $f(x) = \frac{1}{2\Pi\sigma^2} exp\left(\frac{-x^2}{2\sigma^2}\right) - \frac{1}{2\Pi K^2\sigma^2} exp\left(\frac{-x^2}{2K^2\sigma^2}\right)$ describes a DoG filter kernel. In that equation x is the position with respect to the filter origin, σ is the filter's standard deviation and K is a scaling factor tuned to make the DoG best approximate the Laplacian of Gaussian. An example DoG filter is shown in Figure 2.2 with $K = 1.6$ and $\sigma = 1.0$. Lowe's SIFT feature (2004) combines the scale space blob detection with patch orientation based on gradient magnitude and a descriptor based on a histogram of gradients to achieve correspondence between images at up to a thirty-degree change in viewing angle.

Another example of wide-baseline matchable features is the maximally stable extremal region (MSER) first introduced by Matas et al. (2002). Rather than looking for a strong response to a corner or blob detector, MSER features are image regions that have a relatively stable size over a range of different image thresholds. To calculate MSER, the image is converted into a series of binary images with different thresholds where pixels below the threshold are zero and above the threshold are one. Connected components are then found in these binary images. Components that have a relatively constant size over a wide range of thresholds are detected as features. MSER features are invariant to intensity changes and to projective distortion. Hence they are ideal for wide baseline feature matching. Matas et

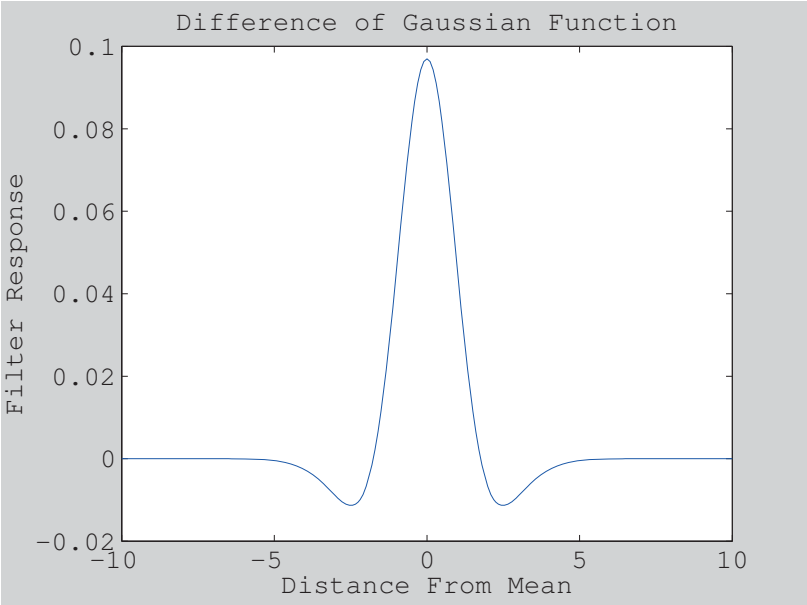

Figure 2.2: A one-dimensional Difference of Gaussian kernel.

al. demonstrated an extraction method that was $O(n \log(\log(n)))$ where n is the number of pixels in the image. Later Nistér and Stewénius (2008) showed that MSER features can be extracted with an algorithm that is at worst $O(n)$.

Each of the preceding feature extraction methods extracts features in each frame indpendently and without regard for the correlations between feature positions that can come from an underlying model of the scene geometry and camera motion. The following section will introduce guided correspondence finding methods that make use of this correlation to improve matching.

2.2.2 Guided Correspondence

Guided correspondence finding uses a camera motion model and inter-feature probabilistic relationships to match features. In its simplest incarnation, active matching can use a prediction of the camera's motion, usually based on a constant rotational and translation velocity model, together with the estimated 3D feature positions to predict the projection of the features in the next image. This approach was used by Davison (2003) to constrain

the search area for correspondences. A refinement is then performed starting at these predicted feature locations to find the features. The advantage of this framework is that it can handle faster camera dynamics than a differential feature tracker, can cope to some extent with repetitive features, and is less computationally expensive than many feature extraction methods.

This search-based approach does not take into account how informative a given feature is about the camera pose. Davison (2005) shows that the mutual information between the map state (3D feature positions and camera pose with uncertainties) and the measurements can determine how much information is gained by finding a given feature. This expected information gain can then be combined with the cost of detecting that feature to determine the relative efficiency in terms of information gain per unit computational cost of detecting a feature. Features can then be found in the most informative order. As each feature is found, it also reduces the uncertainty of the other features, which reduces the computational cost of finding those features. Given a fixed time budget, ordering by relative information efficiency gives the most information about the map state possible.

Chli and Davison show that a mixture of Gaussians model can be used along with the information gain criteria developed in (Davison, 2005) to achieve matching in ”perviously unmanageable cases of jerky, rapid motion,” (Chli and Davison, 2008). While Davison’s previous work was limited to synthetic imagery, Chli and Davison applied their mixture of Gaussians model to a real-time camera tracking 3D SLAM system. However, Chli and Davison’s primary contribution was to show that the correlation between image features could be used to reduce the search area for successive features after initial matches are found. This significantly sped up feature matching over Davison’s standard monoSLAM (Davison et al., 2007) matching implementation based on Joint Compatibility Branch and Bound (Neira and Tardós, 2001). Later work by Handa and Davison (2010) improved on the efficiency of Chli and Davison’s algorithm by sparsifying the dense probabilistic relationships between feature correspondences. With sparsification, Handa et al. can match

many more features in real time than were possible with a dense measurement covariance matrix.

2.3 Relative Pose Estimation

Relative pose estimation is the process of calculating the rotation and translation of a camera or multi-camera system from two sets of images taken at two different times. Relative pose estimation methods can be broken up into two primary classes: those that use sparse feature correspondences as their input, and those that rely on dense optical flow. The former are much more commonly used in practice. An example of the latter, dense optical flow based method, was introduced in (Yang et al., 2007).

Relative pose methods that use sparse feature correspondences can further be broken into two classes, those that use a predictive motion model, and those that do not. A predictive motion model is generally a very simple assumption on the camera motion such as, ”the camera moves with constant velocity.” Filtering approaches to visual SLAM (Azarbayejani and Pentland, 1995; Davison et al., 2007; Eade and Drummond, 2006) use a predictive motion model which, in combination with the expected 3D position of the sparse features based on a static world model and their uncertainties, gives a search region in the current frame for previously mapped features. Once these features are found, the pose estimate can be refined starting from the predicted pose.

The main advantage of a predictive motion approach is that pose estimation does not use random sampling and so the correct relative pose can be found in a fixed amount of time, a necessary property in any hard-real-time system. A primary disadvantage of predictive motion models is that the model used must match the expected motion, or the uncertainty of the predictive model must be accounted for in the filter's process noise. Otherwise, the true feature matches would not be within their predicted uncertainty region and the matching would fail, leading to failure of the relative pose estimate.

Relative pose estimates that do not use a predictive motion model generally rely on random sampling of correspondences between features in the current image and either another image or 3D features in the map itself. This random sampling and consensus (RANSAC) approach was first introduced by Bolles and Fischler in (Bolles and Fischler, 1981). The basic RANSAC algorithm for relative pose is given in Algorithm 1.

Data: Set of putative correspondences C

Result: Relative pose P_i and inlier correspondences C_{in}

confidence in having seen a good solution = 0;

best sample support = 0;

while *confidence in having seen a good solution* $<$ *threshold* **do**

    select a minimal set of correspondences;

    use minimal set of correspondences to calculate relative pose;

    test whether or not correspondences support solution;

    if *current sample support* $>$ *best sample support* **then**

        best sample = current sample;

        best sample support = current sample support;

    end

    calculate confidence in having seen a correct solution;

end

Algorithm 1: The RANSAC algorithm applied to relative pose estimation

Using a minimal set of correspondences to calculate the putative relative pose is desirable because it minimizes the chance that an incorrect correspondence or outlier will contaminate the putative solution. The confidence in the current best solution can be calculated using $c(solution) = 1 - (1 - p_{in}^{\textit{solution set size}})^{\textit{num samples}}$ a variant of which was first introduced in (Bolles and Fischler, 1981). This confidence is dependent on the probability of a correspondence being an inlier p_{in}, the solution set size and the number of samples

taken so far. p_{in} can be taken from the inlier ratio of the best solution found yet in the data. The interplay of the inlier ratio p_{in} and the number of samples determines the number of random samples that must be drawn to generate an output from the RANSAC algorithm with an acceptable level of confidence $c(solution)$. This means that with low inlier ratios, p_{in}, a large number of samples may be required to get a solution with a desired level of confidence. Unfortunately, this means that unless some a priori bound can be given on the inlier ratio, the sampling time is not bounded and so RANSAC cannot be used in a hard-real-time system.

Many optimizations to RANSAC have been suggested (Nistér, 2003; Raguram et al., 2008) but none has overcome this fundamental flaw in RANSAC. Given a fixed time budget these methods will return the least bad solution they can find. However, the confidence in that solution may be too low for the solution to be used in practice if a lower bound on the inlier ratio is not given.

The RANSAC algorithm returns the best minimal sample solution found as well as the set of inliers. To get a more accurate solution, a refinement of the solution should be performed using all of the inlier correspondences. Even the best solution after RANSAC is still based only on a small set of correspondences which themselves contain some amount of noise. Assuming the noise on the inliers is normally distributed, refinement with a larger set of inlier correspondences will arrive at a more accurate result. This refinement can be done with a simple linear method or a more complicated non-linear minimization may be required.

2.4 Direct Solution Methods

Direct solution methods are used to calculate the relative pose of a camera from correspondences in each RANSAC sample. Many different direct solution methods exist to calculate the relative pose of a camera from a sequence of images. Each method is aimed at a spe-

cific problem such as homography estimation (Brown et al., 2007; Horn, 1987), motion estimation for calibrated monocular cameras (Nistér, 2003), uncalibrated monocular cameras (Hartley and Zisserman, 2004), multi-camera systems (Clipp et al., 2008; Kim et al., 2007) or generalized camera approaches (Stewénius et al., 2005).

Each direct solution method is typically formulated as a problem of algebraic geometry and then solved using a unique method of the author's choosing. Recently, Kukelova et al. (2008) have developed a method to automatically generate minimal problem solvers. Careful consideration must be given to numerical performance in a direct solver, as this can greatly influence the reliability of the solver's result. Additionally some solution methods are based on finding the roots of a polynomial function. Each of these roots must be tested in the RANSAC framework as a separate possible solution and so the degree of the polynomial also has a great impact on performance.

This dissertation focuses on visual SLAM for rigid multi-camera systems and so some background on direct solution methods for these systems is in order. Nistér's seminal work on visual odometry (2004) used the perspective three-point (P3P) method (Haralick et al., 1994) to generate relative pose samples. The P3P method uses correspondence between an image from a calibrated camera and a set of 3D point features to find the pose of the camera with respect to those features. In Nistér's case he built a map of 3D point features as he moved the camera, enabling him to estimate the next camera pose with respect to the existing map. His verification in RANSAC was then done with correspondences from both cameras in his system's rigid stereo head.

Nistér also developed a generalized three-point method that could work with a rigid multi-camera system with one to three cameras (Nistér, 2004). Three correspondences between the images and the 3D point features could be drawn from any combination of the cameras to generate the solution with each feature projection having its own center of projection. However, improved results were not demonstrated with respect to the P3P method where all feature projections share the same center of projection.

A particularly challenging problem is to develop a method to calculate the six degree of freedom motion of a non-overlapping multi-camera system. Weng and Huang (1992) presented one of the earliest works in this area. In their work they described how this motion could be calculated using a non-minimal linear equation and described motions that could lead to degenerate solutions algebraically. Their equations allow for the non-overlapping multi-camera system's extrinsic calibration to change between images. However, this calibration must be known at each frame.

Dornaika and Chung (2003) extended the work of Weng and Huang (1992) by showing that a non-overlapping multi-camera system could be calibrated up to an unknown scale factor without stereo correspondence. Their method uses a three-stage process. First, the ego-motion of each camera is calculated separately. These ego-motions are then combined to calculate the relative orientation between the multi-camera system's cameras. Finally, the relative scaling between the motions of the various cameras is found, placing each of the cameras in a single arbitrarily scaled coordinate frame.

Frahm et al. (2004) also developed a pose estimation method for multi-camera systems with non-overlapping fields of view. Theirs is a linear but non-minimal approach. Interestingly, when applying their equations to model a single camera system the equations reduce to the linear single camera pose estimation method with respect to a known set of homogenous world features given in (Hartley and Zisserman, 2004). Frahm et al. also develop a method to automatically calibrate a multi-camera system up to an unknown scale factor from correspondences. In contrast to Dornaika and Chung (2003), the calibration method of Frahm et al. requires overlap in the cameras' fields of view.

Kim and Chung (2006) studied the problem of estimating the six degree of freedom motion (including scale) of multi-camera systems from only temporal correspondences. They give proofs for all of the degenerate motions the camera system can take that do not allow for the scale of the motion to be calculated. These motions will later be introduced in Chapter 3. They then develop an Extended Kalman Filter for 6DOF structure and motion

estimation using the known geometry of the multi-camera system to find the absolute scale of the scene and motion without stereo correspondence. Kim and Chung also point out that adding a simple rotation of the multi-camera system about a single axis as the rig is moved can prevent degenerate motions in almost all practical cases.

Kim et al. (2007) have also solved the problem of 6DOF motion estimation by formulating it as a triangulation problem. They first solve for the rotation of the multi-camera system by averaging the rotations measured by each of the cameras independently. They then triangulate the translation directions of the system's cameras to arrive at a scaled measure of the camera system's translation. They use a second order cone programming approach for triangulation, which is optimal in the L-Infinity norm. Since it requires 5DOF motion estimates for each of the cameras to calculate the scaled motion, theirs is a non-minimal solution method.

Tariq and Dellaert (2004) have also developed a method for estimating the pose of a multi-camera system. Their method uses a non-linear minimization of the reprojection error of known 3D features. Using synthetic data, they show that increasing the number of cameras in a multi-camera system can improve the rotation and translation estimation accuracy, and prevent catastrophic failures due to a lack of tracked features.

Ni and Dellaert (2006) developed a method for six degree of freedom stereo camera motion estimation based on decomposing the problem into estimating the rotation from points at infinity followed by the translation. Their method assumes an initial starting point close to the true motion solution, and then performs a non-linear minimization to find the rotation, and then translation in a two-stage approach. Assuming an initialization as they do is appropriate for applications processing video sequences but cannot be used on general photo collections.

Another approach to solving the 6DOF motion of non-overlapping rigid multi-camera system is the generalized camera framework (Grossberg and Nayar, 2001; Pless, 2003). A generalized camera is an image formation system, which is made up of a set of rays. These

rays can intersect at any number of optical centers with each ray possibly having its own optical center. A generalized camera can be used to model any camera or multi-camera system. Nistér's (2004) generalized three-point solution method is based on a generalized camera model, hence its name.

Stewénius et al. (2005) presented a minimal solution method for estimating the six degree of freedom motion of a generalized camera from image correspondences. They showed that there are up to sixty-four solutions to the relative pose of two generalized cameras given six ray correspondences. One of the limitations of their approach is that it is degenerate for generalized cameras where the rays' centers of projection are all on a line. This naturally excludes all two-camera systems, as two camera centers always form a line.

Chapter 3 will describe one of the contributions of this dissertation, a novel six degree of freedom relative pose estimation method for a non-overlapping rigid two camera system using a minimal set of six correspondences. Using a system of two or more non-overlapping cameras, scene coverage can be maximized with this method while still measuring the true scale of the motion. In Chapter 4, another contribution of this dissertation is described. This is a method for estimating the scaled, six degree of freedom motion of a stereo camera with only a small overlap in the cameras' fields-of-view. This novel method overcomes some of the limitations of the method introduced in Chapter 3 while still giving the multi-camera system a large total field of view.

Each of the processes described to this point can operate with only temporally or spatially local information, including feature extraction, feature tracking, and relative pose estimation. The next section will give background on global mapping operations, which use non-local information to create a map of the camera system's operating environment.

2.5 Global Mapping

In this dissertation, global mapping refers to the processes that a SLAM system must perform to create an internal representation of its operating environment when all of the environment is not visible from a single point of view. At a minimum, the global map must reflect the topology of the operating environment, i.e. the connectivity between the various parts of the environment. This requires that the VSLAM system perform *loop detection*.

However, for many practical applications, a topological map is not enough and a Euclidean map is required. Some example applications where a Euclidean map is required are efficient path planning and automated targeting. A Euclidean map is the sort of map we are generally accustomed to where each point on the map is known with respect to every other point and the map has a single scale. Additionally, in a Euclidean map distances and angles are preserved. When creating a Euclidean map it is not simply enough to make note of a loop when it is detected. The system must also perform *loop correction* to make the geometry of the map reflect the Euclidean geometry, not just the topology, of the operating environment. Additionally, the global geometry of a Euclidean map can help to eliminate false loop detections. For example, when mapping a building by moving around it, if two corners of a building look the same and have similar geometry, we can use the shape of the path taken around the building to disambiguate the corners.

2.5.1 Loop Detection

Visual loop detection has been studied extensively in the vision and robotics communities. Some authors chose to tackle loop detection based only on appearance (Cummins and Newman, 2008; Eade and Drummond, 2008; Nistér and Stewénius, 2006) while others included geometry in their approaches (Chli and Davison, 2008; Davison, 2003; Eade and Drummond, 2006; Ankur Handa and Davison, 2010; Irschara et al., 2009; Sivic and Zisserman, 2003; Williams et al., 2007).

Sivic and Zisserman (2003) were the first to apply text retrieval techniques to find objects within a video sequence. Their approach is suitable for loop detection since it finds images with common structure in a video sequence. They use both SIFT descriptors (Lowe, 2004) and maximally stable extremal regions (MSER) (Matas et al., 2002). Their approach consists of a preprocessing step followed by a retrieval phase. In the preprocessing step SIFT and MSER features are extracted in each of the images. A subset of these image's features are then grouped using K-means clustering. Each of these clusters is referred to as a visual word. They use approximately 500 images in the clustering and generate approximately 10000 distinct visual words. They are limited in the number of clusters they can generate by the complexity of K-means clustering with such a large number of clusters. Each image's features are then quantized into visual words. An inverted file is stored for each visual word, which contains which images a visual word is found within and the visual word's position in that image.

When querying to find an object in the video the user selects a region of interest containing this object. The system then finds the descriptors in this region and their visual words. It then uses a term-frequency inverse document frequency (TF-IDF) weighting formula to find other images in the video that contain the same visual words. The TF-IDF weighting takes into account both the frequency of a given visual word in the current image and the log inverse frequency of a visual word in the documents. The TF-IDF formula is $t_i = \frac{n_{id}}{n_d} \log \frac{N}{n_i}$ where n_{id} is the number of occurrences of visual word i in document d, n_d is the total number of visual words in document d, N is the number of documents in the database and n_i is the number of documents containing visual word i.

Each document in the database (video) is represented by a vector of TF-IDF values for each visual word:

$$v_d = (t_1, ..., t_i, ..., t_v). \tag{2.1}$$

The dot products of the vector representing the query region and each document in the database can then be taken to find images that are likely to contain the object of interest. The method then performs a final geometric verification step. Their method starts with two matched feature regions and looks at the local neighborhood of each feature. The same features in the local neighborhood of the matched features should be found in both frames. This is a form of loose geometric verification, which does not enforce any sort of global model such as a homography or essential matrix.

Sivic and Zisserman show that their system vastly out-performs this naive approach in terms or run time as well as precision-recall curves. The improvement in precision and recall is primarily due to the TF-IDF weighting of the descriptors and their weak geometric verification.

Nistér and Stewénius (2006) extended the work of Sivic and Zisserman in image retrieval. Their vocabulary tree approach uses a hierarchical k-means clustering of the feature space to partition features into visual words. The tree-based approach overcomes scaling issues that limited the vocabulary size in Sivic and Zisserman (2003). With a larger number of visual words, the descriptor space can be broken up into more discriminative, smaller clusters. A larger number of smaller clusters improve image retrieval performance. Additionally, because their visual words were more discriminative, Nistér and Stewénius found that they did not need to use geometric verification to achieve good performance. They show results on up to a one-million image database, three orders of magnitude larger than Sivic and Zisserman's results.

Cummins and Newman (2008) developed a probabilistic approach to loop detection using only image information without geometric verification. Their fast appearance based mapping (FAB-MAP) approach uses a bag-of-words (Sivic and Zisserman, 2003) approach to represent the images where the features are quantized into particular visual words. However, rather than use TF-IDF to determine the likelihood of two images matching, Cummins and Newman learn a generative model for the visual words. This model, learned from train-

ing data, reflects the fact that certain visual words are likely to co-occur in an image because they come from the same object. Knowing which features are likely to appear or disappear together because they are part of the same object or objects, the system in effect recognizes unique locations by finding unique groupings of objects in the imagery. The system also recognizes which visual words appear in many locations because of repetitive scene parts. This is known as perceptual aliasing. In Nistér and Stewénius's approach, perceptual aliasing is handled by the inverse document frequency term while in Cummins and Newman's model a probabilistic approach is taken.

Cummins and Newman make a comparison between a naive Bayesian assumption that the likelihood of a visual word being found in an image is independent of the other visual words in the image and a model that takes into account the correlations between visual word occurrences. They show that using a simplification of the full correlation between all features based on a Chow Liu tree (Chow and Liu, 1968) can provide significant performance improvements over the naive Bayesian approach at little additional computational cost. The Chow Liu tree is a maximal spanning tree over the correlation between features which, while it is a simplification over the full correlation matrix, provides good performance with close to real-time computation.

Loop detection methods that use geometry as well as appearance can be divided into two classes: those that first use a camera pose prior and scene geometry to guide appearance based matching (Chli and Davison, 2008; Davison, 2003; Eade and Drummond, 2006; Ankur Handa and Davison, 2010), and those that first match based on appearance and then verify the match geometrically (Irschara et al., 2009; Wang et al., 2005; Williams et al., 2007). Geometry-first approaches presume that there is some non-uniform prior on the camera's pose within the map and use this prior to guide feature matching. The mean and variance of the 3D features are projected into the next expected camera. In combination with the uncertainty on the current camera pose, this gives a search region in the image to look for each feature. Chli et al. showed that when sequentially finding features in this

way, the information from one feature match can reduce the pose uncertainty and therefore limits the search region size for subsequent features. In later work, they developed more efficient methods to represent the probabilistic constraints between the feature projections (Ankur Handa and Davison, 2010). These methods make real-time probabilistic feature matching possible.

Geometry-first methods have been successfully demonstrated on small room-sized environments. However, were they to be used in mapping larger environments, the large uncertainty in the camera pose prior after traveling long distances would become an issue. With a large pose uncertainty, such as one would have after traversing a large loop, the projection of the feature uncertainty regions in the image would be extremely large. Also, the number of features that might project in the image grows as the camera may take on a wide range of poses. A more efficient approach is needed.

Appearance-first approaches can detect loops of any size because they do not rely on a camera pose prior. In the work of Irschara et al. , a vocabulary tree (Nistér and Stewénius, 2006) and inverted files are used to find visually similar parts of a database of images. These images have been previously processed into a 3D point cloud model of the environment. Features are then matched between the query image and the map based on descriptor similarity (dot product of SIFT features). The perspective three-point method (Haralick et al., 1994) in a RANSAC framework is then used to verify that the geometry of the scene could indeed generate the feature distribution in the query image. However, Irschara et al. do more than just this. They create virtual views of the 3D point cloud model and use these as the database that query images are compared with. This significantly reduces the number of images in the database, speeding up the histogram generation from the inverted files. They also make extensive use of the graphics processing unit (GPU) to speed up processing query image features into visual words through the vocabulary tree.

A wholly different approach to feature or key-point identification was developed by Ozuysal et al. (2007,2010) and demonstrated in a real-time monocular SLAM system by

Williams et al. (2007). Their approach, which they call ferns, uses a collection of simple Bayesian classifiers to separate key-points into a set of classes. These classifiers (ferns) start with an image patch representing the key-point. Each classifier is then made up of a set of binary decisions. Two pixels are randomly selected in the patch and if one pixel is brighter than the other the value is one, otherwise the binary decision value is zero. Using a large number of these random binary measurements over the image patch a distribution of resulting binary numbers (fern results) can be found over a large training set of examples of the same image patch. These examples can be synthetically generated affine warps of an original image patch. In practice the number of binary decisions (leaves in the fern) must be large (L = 300) to achieve acceptable classification performance. For each key-point, the method must calculate the probability that it takes on any one of the 2^{300} possible fern output values. This is intractable and so the authors simplify the computation by breaking up the large fern into M smaller ferns each of which can take on $2^{L/M}$ values. The classification results of this group of ferns are then multiplied together to form the final probability of a feature being of a particular class. Grouping the binary decisions into small ferns performs better experimentally than assuming that each of the binary decisions is statistically independent.

Ferns were an outgrowth of and an improvement on Lepetit and Fua's work on randomized forests (Lepetit and Fua, 2006). Lepetit and Fua were the first to consider the wide-baseline matching problem as a classification problem where a key-point should be mapped to a class or view-set. This is the same approach they later took in Ferns but with the difference that randomized forests make use of randomized trees (Amit et al., 1996) for the classification while ferns makes use of the non-hierarchical fern structure.

Ozuysal et al. 's ferns perform the same sort of quantization as a combination of affine invariant feature extraction (SIFT for example) followed by hierarchical quantization through a vocabulary tree. However, ferns do not require an affine invariant feature to be extracted with a descriptor vector, which is usually costly to compute. Also, the number

of ferns and the number of binary decisions in each fern can be tailored to the computational resources available on a given platform. Fewer ferns might be used in an embedded system while more might be used in a laptop for example. Of course, this comes at a price. With fewer ferns, the features can be broken into fewer classes reducing the classes' discriminative power.

Williams et al. (2007) showed that ferns could be used to recover from a loss of tracking in monocular VSLAM. While loss of tracking is not exactly the same problem as loop detection, Eade and Drummond (2008) have pointed out that the same location recognition mechanism can be used for both. Willams's SLAM system is based on an Extended Kalman Filter for map and pose estimation (Davison et al., 2007). Each feature that is added to the map is passed to a background process that warps a patch about the feature in various ways such as rotation, scaling, and perspective warping. These warped patches are then passed into the ferns which learn the distribution of appearances that may occur for a given feature patch. Later when the camera becomes lost, the system can classify the features in a new image using the ferns to find likely matches between the features in the current image and those in the map.

Since their map only contains on the order of eighty features they have no need to use an inverted file data structure to find likely matching images to the current image, but can simply do exhaustive matching to the features in the map. However, they do cull the possible feature matches based on prior common visibility (the features were all seen together before) and proximity. Features that are not close together in the map are not likely to be correctly matched to the features in the current image. Finally, Williams et al. use the perspective three-point method in a RANSAC procedure to geometrically verify their matches.

2.5.2 Loop Correction

After a loop is detected, some process must be performed to correct the accumulated error in pose estimates and scene structure over the camera's path. A given system's approach to loop correction depends on the kind of map that the SLAM system builds. A system may create a Euclidean map, one in which all points in the map are known with respect to each other and the shortest distance between two points is a line, or a topological map which models the connections or transitions between various locations, but does not give an overall picture of the shape of the environment.

An atlas is an example of a Euclidean map we have probably all used at some point. The map has a single common scale that can be used to find the distance between any two points on the map. On the other hand, a subway map is a common example of a topological map. A subway layout shows the way to get between a set of stations along a prescribed set of paths. However, it does not show the exact geometry of the subway system.

Euclidean maps are more common in SLAM systems because they allow paths to be planned through the map which have not been traversed before and because they naturally represent the Euclidean structure of the world we inhabit. However, this universal knowledge of location comes at a high computational cost. Topological mapping systems, because they do not have a Euclidean structure, can be much less costly. A topological map at its simplest is a graph with nodes representing locations and edges representing known paths between locations. Loop correction in a topological map is as simple as adding another edge to the graph. This makes topological maps an attractive alternative in real-time systems. Recently, hybrid approaches have also been developed which are locally Euclidean but globally topological in nature (Sibley, 2009; Sibley et al., 2009).

2.5.3 Globally Euclidean Loop Correction

Two opposite ends of the global loop correction spectrum are taken up by Extended Kalman Filter based approaches and bundle adjustment. In EKF approaches, the map and only the latest camera pose are corrected after a loop is detected. In contrast, bundle adjustment tries to create a maximum likelihood estimate of all camera poses and the map. Sub-map based approaches fall in-between these two ends of the spectrum. They generally break the map into a set of many sub-maps. Each of these maps is then corrected separately in their own coordinate frames. Lastly, the sub-maps are held internally fixed and a correction process is performed which minimizes the error of measurements between sub-maps by changing the sub-maps' relative poses.

Extended Kalman Filter approaches to VSLAM such as Davison's (2007) only estimate the most recent camera pose. After detecting a loop, the current camera pose and map are updated to reflect the new measurements. If one were to plot the camera's path based on the EKF estimates, one would see a discontinuity in the camera's pose just after loop completion, since the previous poses are not updated when the loop is found. This will cause significant problems for procedures that rely on an accurate camera path, including dense stereo matching and scene reconstruction.

Bundle adjustment (Triggs et al., 2000) represents the opposite end of loop correction spectrum. Using a non-linear minimization, bundle adjustment seeks to find the globally optimal camera path and 3D scene structure given the images in a video sequence and feature correspondences between the frames. Many different parameterizations of bundle adjustment exist, but the basic form of the bundle adjustment error term is

$$\sum_i \sum_j d(x_{ij}, \Pi(R_i, C_i, X_j))^2 \tag{2.2}$$

where i is the camera index, j is the 3D feature index, R_i is the rotation of camera i, C_i is the center of camera i, X_j is the three-dimensional position of feature j, x_{ij} is the

measured value of feature j in camera i, Π models the projection of the 3D point into the camera, and $d()$ is a measure of the distance between the measured and expected projection of the feature and is generally given in image pixels. Typically, bundle adjustment procedures minimize this sum of squared errors using a Levenberg-Marquardt algorithm (Levenberg, 1944; Marquardt, 1963). However, others have shown promising results using preconditioned conjugate gradients (Byrod and Astrom, 2009; Dellaert et al., 2010).

Bundle adjustment is a highly sparse minimization problem. Each expected measurement is only affected by a single camera and 3D point. This gives the Jaccobian of the projection function Π its sparse structure. Lourakis and Argyros (2009) take advantage of this structure in their open source implementation sparse bundle adjustment, or . The bundle adjustment problem state can be partitioned into cameras and points. The Schur complement can be used to factor out the points, converting their non-zeros in the full Jaccobian matrix into constraints between cameras that both see the same feature. The features are generally factored out rather than the cameras because there are many more features than cameras in a typical bundle adjustment problem. However, in a situation where a camera moves in the same environment for a long period of time, it may be more efficient to factor out the cameras if there are more camera poses than points. At each iteration of the minimization, the Schur complement is used to factor out the features, the camera portion of the state is updated based on the measurement residuals, and finally the updated 3D feature positions are calculated.

Bundle adjustment re-linearizes the measurement functions in each update step. In contrast, the EKF linearizes the measurements only once about the state estimate of the feature and camera at the time of the measurement. The influence of a given measurement is then folded into the EKF's pose and map estimates as well as their uncertainties. The problem with this approach is that if the linearization point was far away from the true position the EKF will not be able to recover from this error. In contrast, starting from a state somewhat far from the global minimum of the bundle adjustment cost function

(Equation 2.2), re-linearization can allow bundle adjustment to converge to the correct solution where an EKF will not. Of course, the error function has local minima in which the optimization may get stuck.

In addition to problems with local minima,due to its least squares formulation, bundle adjustment cannot deal directly with outlier measurements. However, a lack of robustness is a problem for any least-squares framework, including the EKF. Robust bundle adjustment methods try to deal with outliers by down-weighing their influence in the minimization. A simple method to do this is to assign a higher uncertainty to measurements that exhibit large residual errors (McGlone et al., 2004). Other approaches involve down-weighing the error term for measurements with large residuals according to some function (McGlone et al., 2004).

Bundle adjustment can be extended to other types of sensors besides cameras. Thrun et al. describe the GraphSLAM algorithm in their book Probabilistic Robotics (Thrun et al., 2005). The GraphSLAM algorithm generalizes bundle adjustment to include robotic control inputs, odometry information, and any other type of sensor information that can be used to map a pose of the robot or camera to another pose or a pose to a feature in the world.

2.5.4 Loop Completion by Subdivision

While bundle adjustment is the gold standard for loop completion methods, its computational complexity scales with the cube of the lesser of the number of cameras or features in the map. This makes bundle adjustment impractical for large loop closures that have to be done in real time. Various methods exist to deal with this complexity problem. Most fall into the category of divide and conquer or hierarchical approaches.

Fitzgibbon and Zisserman (1998) developed one of the earliest hierarchical scene reconstruction methods for image sequences. Their technique breaks the sequence into se-

quential groups of three images. These image triplets are then reconstructed independently, finding image correspondences and a trifocal tensor between the images. This yields a large set of projective reconstructions. Each of these projective reconstructions is then bundle-adjusted independently. They are then combined by estimating a homography of three-space between the two projective reconstructions and bundle-adjusting the homography. This same homography estimation and refinement process can then be applied hierarchically to join the entire sequence. A final bundle adjustment on all views is then performed to create the resulting reconstruction of the total sequence.

Shum et al. (1999) take a different approach to hierarchical bundle adjustment. They split a video sequence into many small sections which they independently reconstruct using bundle adjustment. Then they select two *virtual key frames* for each section. One virtual key frame could be the first frame in the sub-sequence and the other is selected at some other location. They calculate the 3D uncertainty of the features based on the measurements and cameras in a subsequence. This uncertainty is projected into the two virtual key-frames and stored. This gives two frames for each sub-sequence that contain all of the information constraining the other camera poses and the features up to linearization error.

The sub-sequences are then joined together into a single model with only the two virtual key-frames included in the final reconstruction for each sub-sequence. In the final reconstruction, the uncertainty of the 3D points projected into the virtual key-frames is modeled by non-isotropic measurement errors on the virtual measurements. The use of virtual key-frames dramatically reduces the number of images in the final reconstruction, achieving faster processing speed in the final bundle adjustment. At the same time, the virtual measurements with their associated non-isotropic uncertainties contain all of the information about the structure in the original images.

Nistér (2000) extended the work of Fitzgibbon and Zisserman (1998) and dealt with the problem of view selection. While Fitzgibbon and Zisserman assume that all sequential sets of three images can be used to create accurate trifocal tensors, Nistér developed a method

for selecting sets of views that were more likely to work well. In view selection, there is a tradeoff between the amount of parallax feature correspondences exhibit, which increases over time and the number of correspondences between images in a sequence, which tends to decrease over time. Nistér's approach attempted to find the sweet spot in this tradeoff that results in the best reconstruction possible with general amateur videos, which may have varying camera separation over time. The view selection criterion measures both the number of feature correspondences and the ratio of correspondences that are inliers to the trifocal tensor but not to a homography. Effectively, this is a measure of the parallax in the scene. More parallax is desirable to arrive at an accurate 3D reconstruction. In addition to finding good view triplets, Nistér also used line matches in his system to improve the reconstruction accuracy. The guided line matching approach used can be found in (Schmid and Zisserman, 1997).

How to break up a collection of cameras and feature correspondences to speed up bundle adjustment was further studied by Steedly et al. (2003). Their approach was to break the graph of cameras and features into weakly connected sub-maps and then bundle-adjust these independently. The sub-maps are then fixed internally and optimized with respect to each other to arrive at the final reconstructed cameras and scene geometry. They used spectral graph partitioning on the Hessian matrix to find the low-error modes of the reduced Hessian with the 3D features factored out using the Schur complement. Their partitioning took into account the fact that each camera has multiple corresponding rows and columns in the Hessian and forces the partitioning to keep each camera's parameters together in a single partition. They compared the reprojection error after their spectral partitioning method against a simpler partitioning approach based only on the non-zero entries of the Hessian. This simpler approach does not take into account the degree of connectedness between different sets of cameras but only whether or not two cameras measure any common features. They show that by using a partitioning in the low-error modes of the camera system, they can achieve lower final reprojection errors than the simpler visibility based approach.

Chli and Davision presented a method to split a map of camera poses and cameras hierarchically based on the mutual information between expected feature measurements in (Chli and Davison, 2009). Using the mutual information between measurements, features can be clustered into groups to split the map into sub-maps. These sub-maps can be further split in a hierarchical fashion and independently optimized, then held internally fixed and optimized with respect to each other.

Ni et al. (2007a,2007b) developed an out-of-core bundle adjustment method for large-scale 3D reconstruction. Their method starts by decomposing the graph of cameras and features in the bundle adjustment into a set of sub-maps. This partitioning is done with a graph cut that minimizes the number of edges (visibility connections between cameras and 3D features) that span the sub-maps. Each of these sub-maps is parameterized independently with one camera of each sub-map serving as the base node or origin of the sub-map's local coordinate frame. This allows the sub-maps to be optimized separately. Their method partitions the measurements in the reconstruction into those that depend on multiple sub-maps and those that depend only on a single sub-map. Cameras or features that yield measurements that only depend on a single sub-map are *internal variables* while those that span different sub-maps are *separator variables*. First, the internal variables are optimized for each sub-map. This can be performed in parallel. The linearizations of the internal variables are then cached and used in optimizing the separator variables while holding the internal variables fixed. Finally, the separator variables are held fixed and the internal variables are given a final polish. This separation mechanism reduces the total time required to reconstruct a sequence even without using parallel processors because the Cholesky factorization used in each iteration of bundle adjustment is super-linear in its complexity.

Rather than a hierarchical approach, Snavely et al. developed a method that selects the most important cameras in a bundle adjustment and only bundle adjusts the cameras in this *skeletal set* (Snavely et al., 2008). After bundling the skeletal set, they could then fix this

set and the associated 3D geometry and use standard robust pose estimation techniques to find the pose of the cameras not in the skeletal set. Their algorithm increased bundle adjustment speed by more than an order of magnitude with ”little or no loss in accuracy” (Snavely et al., 2008).

The skeletal set is built by first creating a graph with edges between every pair of cameras that share features in common. The edge weight contains the trace of the covariance matrix representing the uncertainty of the relative pose between the two cameras. This graph is then pruned, removing edges, checking that removing an edge in the graph does not increase the relative uncertainty between any two nodes in the graph by more than a factor t. The relative pose uncertainty between any two nodes is measured by summing the covariance trace values along the shortest path between the two nodes. Snavely et al. developed an efficient algorithm based on a maximum leaf spanning tree (Guha and Khuller, 1998) to prune edges and arrive at the skeletal set. This skeletal set contains considerably fewer cameras than the original graph for typical Internet photo collections, which are the application of this work. Although not an exact measure, having fewer cameras in a bundle adjustment is very likely to increase processing speed.

Since image sequences were not the target of the skeletal set approach, there is still some question as to whether the skeletal set approach would dramatically improve loop correction speed in visual SLAM. However, this appears to be a promising possible approach since it would eliminate the many redundant cameras in a video sequence, increasing processing speed.

Frahm et al. (2010) developed an approach to efficient bundle adjustment that might be applied to loop correction. Their method clusters images based on appearance and selects a single image from each cluster as a representative of the cluster or *iconic image*. Only the poses of the iconic images are corrected in the bundle adjustment rather than the entire set of camera poses.

Eade and Drummond presented a state of the art monocular SLAM system in (Eade

and Drummond, 2008) which uses many local coordinate frames to store the map and camera poses. Their system will be described in detail so that their unified loop correction and tracking recovery mechanism can be introduced. A many-coordinate-frame approach was first put forward by Bosse et al. in their Atlas framework (Bosse et al., 2004). In contrast to Bosse et al. who used the number of features in a given sub-map as a measure of when to create a new sub-map, Eade and Drummond make note of the non-linearity of the projection of a feature into a camera and use this as the basis to decide when to create a new local coordinate frame or sub-map. In Eade and Drummond's system, features can exist in many sub-maps and are stored in inverse depth parametrization (Montiel et al., 2006) with an associated information matrix. The inverse depth parametrization has the advantage that it is nearly linear close to the coordinate system origin, which in this case is the local coordinate frame's origin. The projection of a feature with standard parametrization into a camera with no rotation and translation T is:

$$f(x) = \pi(x + T)$$

$$= \frac{1}{z + T_3}(x + T_1,\ y + T_2)$$

Note that even for $T_3 = 0$ the projection is non-linear in the feature's coordinates. However, using an inverse depth representation where the features coordinates are:

$$(u\ v\ q)^T \equiv \frac{1}{z}(x\ y\ 1)^T$$

This makes the observation model near the origin nearly linear:

$$
\begin{aligned}
f((u\ v\ q)^T) &= \pi((\frac{1}{q}(u\ v\ 1)^T) \\
&= \pi((u\ v\ 1)^T + qT) \\
&= \frac{1}{1+qT_3}(u + T_1\ v + T + 2)^T
\end{aligned}
$$

Eade and Drummond use an iterative minimization to incorporate new measurements into a node and estimate the new camera's coordinates. The near-linearity of the inverse depth parametrization for features makes the minimization converge very quickly, they say more than 99% of the time in one iteration. The near-linearity of the feature parameterizations depends on the current camera being close to the origin of the current node. They use the trace of the Hessian of a synthetic measurement with coordinates $(0\ 0\ 1)$ in the current node to measure the non-linearity of the current camera in its node. If the non-linearity is too high, they move the camera to a different node that is more linear or create a new node.

In Eade and Drummond's system each local node is connected to its neighbors (nodes it shares features with) using a similarity transform. Common features between any two nodes introduce constraints between those nodes. Eade and Drummond update the global map by finding loops in the graph of local nodes connected by similarity transforms. They then use a non-linear minimization to update the similarity transforms between nodes. Since any similarity transform cycle in the graph should sum to identity, they use a minimal spanning tree to find the cycles. Any transform connecting two nodes of the spanning tree along and edge not in the spanning tree introduces a cycle in the graph. A cost function is used that penalizes the divergence of any transform in the cycle from the transform that the inter-node feature constraints support. This cost function over all cycles is then minimized to update the global graph of local coordinate frames. Their method of converting inter-sub-map measurements into virtual measurements constraining the sub-maps is very similar to the FrameSLAM work by Konolige and Agrawal (2008) which will be discussed

below.

The use of cycles in the graph of poses or local coordinate frames for global map correction is becoming a topic of interest in VSLAM. Zach et al. (Zach et al., 2010) use cycles in the graph of camera poses in bundle adjustment to detect incorrect feature matches between images. These incorrect correspondences might be due to repetitive structure in the scene as suggested by Zach, but may have other causes as well. By considering cycles in the graph in a Bayesian framework, they can determine which relative pose estimates between a pair of cameras are likely to be incorrect and prune the measurements connecting these cameras from the bundle adjustment before starting the minimization. Removing these gross outliers dramatically improves the quality of the bundle adjustment results when repetitive structures are present in the scene.

One final approach to loop completion that will be covered here is FrameSLAM by Konolige and Agrawal (2008). In their approach, feature correspondences are found only between pairs of images, rather than across many images. For each pair of images with correspondences in common, a transformation between the cameras is calculated with an uncertainty found by marginalizing out the 3D features measured in both images. This camera-to-camera transformation with uncertainty is then used as a synthetic measurement in a non-linear minimization over all camera poses. The advantage of this approach is that it factors out the 3D features from the non-linear minimization once and for all and so saves computation over standard sparse bundle adjustment. Unfortunately, because Konolige and Agrawal only use two-frame matches to estimate the 3D features, the linearization point they use to calculate the relative pose and uncertainty between the pair of cameras can be inaccurate in comparison to methods using longer feature tracks. This can lead to inaccurate transformations, which are inconsistent, meaning they have greater certainty than they should because of linearization error. These measurements are not re-linearized as they would be in an update step in bundle adjustment. This can lead to inconsistency in the entire map and inaccuracy in the camera path.

With measurements of features converted to synthetic measurements between poses, their approach can then further marginalize out camera poses using the Schur Complement. This creates synthetic measurements between camera poses that did not share any features in common and reduces the size of the state in the non-linear minimization. This state reduction dramatically speeds up the minimization but at the cost of some accuracy. By eliminating cameras from the map, the authors have a VSLAM system that creates maps that scale with the size of the environment rather than the number of key-frames used to measure the environment. This is a very desirable property for any SLAM system.

Unfortunately, factoring out a camera in their framework generates measurements between all of the other cameras that share a synthetic measurement with the removed camera. This is not a problem when synthetic measurements are only allowed for cameras that are temporal neighbors in the video sequence. However, when loops are completed, or if longer tracks are allowed, the reduction in the state size through factoring out cameras can cause an explosion in the number of synthetic measurements, slowing the minimization process.

Performing loop correction in real-time remains an open research problem in computer vision and robotics. Each of the approaches mentioned above has its advantages. However, none has been shown to scale to large enough environments to support loop correction for mobile robots operating in city scale environments. Since loop correction only needs to keep up with the exploration speed of a robotic platform, real-time loop correction in city-scale environments is probably as fast as loop correction ever needs to be for practical purposes. At the moment, solutions that could scale to cities do exist but only for topological maps. These will be introduced next.

2.5.5 Hybrid Metric-Topological Loop Completion

Topological maps have been used in SLAM for many years (Angeli et al., 2008; Kuipers, 1978; Werner et al., 2009). Since they do not support path planning based on Euclidean

constraints, there is some question about their usefulness in robotics applications. Additionally, purely topological maps cannot be used for augmented reality applications. An example of a purely topological map is a graph with nodes representing locations each of which may contain many images, and with edges representing a temporal connection between areas (Angeli et al., 2008). This temporal connection implies that two locations are close to each other, since they have been traversed in close succession.

However, recent advances have been made in hybrid Euclidean-topological mapping that resolve some of the problems with topological maps and Euclidean maps. In hybrid Euclidean-topological maps, the geometry at any one point in the map is known and each point in the map (possibly a camera pose or key-frame) is connected to other nearby points in the map by geometric transformations. Sibley et al. 's *Relative Bundle Adjustment* (Sibley, 2009) introduced this sort of map configuration to VSLAM. In contrast to global bundle adjustment approaches that parameterize features with respect to the global coordinate frame, in relative bundle adjustment, feature points are parameterized with respect to the first camera that measures them. Cameras are parameterized in relative coordinates with respect to each other. Each key-frame in a video sequence is parameterized with respect to some previous key-frame with which it has measured the largest number of common landmarks.

The advantage of hybrid mapping is that by dropping the constraint that the map must be globally Euclidean, loop correction becomes very simple. Loops are corrected by simply adding a new transformation to the graph of camera poses or local coordinate frames. A hybrid map contains all of the measurement information that a Euclidean map does and so at any point using standard bundle adjustment techniques it could be upgraded to a Euclidean map. However, this upgrading does not have to be done on a global scale. If a path needs to be planned in the robot's local environment only the local environment needs to be upgraded to a Euclidean map. The remainder of the map can remain topological. Additionally, as is suggested in (Holmes et al., 2009), the robot's local environment can

be represented with relative coordinates while at the same time the rest of the map can be globally corrected using bundle adjustment in a seamless framework.

This discussion of VSLAM has shown that the VSLAM problem can be broken into many sub-problems. Later Chapters in this dissertation will address the sub-problem of 6DOF pose estimation for multi-camera systems as well as exploiting the subdivisions of the VSLAM problem introduced here to parallelize VSLAM and map office-build scale scenes online and in real-time.

CHAPTER 3

Scaled Motion with Non-Overlapping Two-Camera Systems

3.1 Introduction

Multi-camera systems are an attractive alternative to single cameras or omnidirectional cameras for VSLAM, where the absolute, scaled ego-motion must be calculated. These systems have also been used to capture ground based and indoor data sets for 3D reconstruction (Akbarzadeh et al., 2006; Uyttendaele et al., 2004). They are relatively inexpensive and provide wide scene coverage while at the same allowing scaled ego-motion measurement.

In contrast, with a single camera absolute scale cannot be determined. Omnidirectional cameras based on parabolic mirrors provide wide-angle scene coverage but at the expense of uneven sampling of the visual sphere and also unknowable absolute scale. Some omnidirectional cameras are essentially small clusters of standard perspective cameras (ImmersiveMedia,Imove,Ladybug2) but without a large baseline between the cameras, these cannot be used to measure absolute, scaled ego-motion. This is a problem of signal to noise ratio where the baseline of the cameras is small relative to the scene depth making the scale constraint from the camera geometry weak. Another possible approach to scaled motion estimation from video was presented by Scaramuzza et al. (2009). They use the fact that the camera moves with nonholonomic motion in a plane to calculate the scaled camera motion

Figure 3.1: Example of a multi-camera system on a vehicle

with one point correspondence.

It can be difficult to avoid losing part of the field-of-view due of a single camera or omnidirectional camera due to occlusion, which may require camera cluster placement high up on a boom. Alternatively, for mounting on a vehicle the system can be split into two clusters so that one can be placed on each side of the vehicle and occlusion problems are minimized while giving a large baseline for scale estimation. In this chapter we will show that by using a system of two camera clusters, consisting of one or more cameras each, separated by a known transformation, the six degrees of freedom (DOF) of camera system motion, including scale, can be recovered.

An example of a multi-camera system for the capture of ground-based video is shown in Figure 3.1. It consists of two camera clusters, one on each side of a vehicle. The cameras are attached tightly to the vehicle and can be considered a rigid object. This system is used for the experimental evaluation of our approach.

Computing the scale, structure and camera motion from video of a general scene is an important application of our scale estimation approach. In Nistér et al., 2004 Nistér et al. investigated the properties of visual odometry for single-camera and stereo-camera systems. Their analysis showed that a single camera system is not capable of maintaining a consistent scale over time. Their stereo system is able to maintain absolute scale over

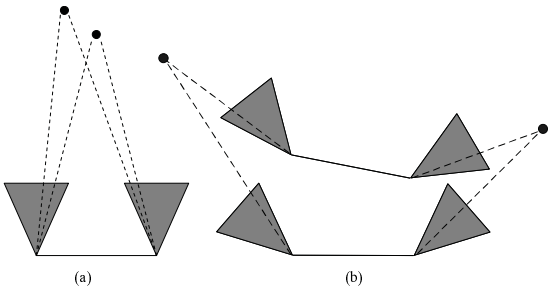

Figure 3.2: (a) Overlapping stereo camera pair, (b) Non-overlapping multi-camera system

extended periods of time by using a known baseline and cameras with overlapping fields of view. Our approach eliminates the requirement for overlapping fields of view and is able to maintain the absolute scale over time.

In the next section 3.2, we introduce our novel solution to finding the 6DOF motion of a two-camera system with non-overlapping views. We derive the mathematical basis for our technique in section 3.3 as well as give a geometrical interpretation of the scale constraint. The algorithm used to solve for the scaled motion is described in section 3.4. Section 3.5 discusses the evaluation of the technique on synthetic data and on real imagery.

3.2 6DOF Multi-camera Motion

The proposed approach addresses the 6DOF motion estimation of multi-camera systems with non-overlapping fields-of-view. Most previous approaches to 6DOF motion estimation have used camera configurations with overlapping fields of view, which allow correspondences to be triangulated simultaneously across multiple views with a known, rigid baseline. Our approach uses a temporal baseline where points are only visible in one camera at a given time. The difference in the two approaches is illustrated in Figure 3.2.

Our technique assumes we can establish at least five temporal correspondences in one of the cameras and one temporal correspondence in any additional camera. In practice, this assumption is not a limitation, as a reliable estimation of camera motion requires multiple

correspondences from each camera due to noise.

The essential matrix which defines the epipolar geometry of a single freely moving calibrated camera can be estimated from five points. Nistér proposed an efficient algorithm for this estimation in (Nistér, 2003). It delivers up to ten valid solutions for the epipolar geometry. The ambiguity can be eliminated with additional points. With oriented geometry the rotation and the translation up to scale of the camera can be extracted from the essential matrix. Consequently, a single camera provides 5DOF of the camera motion. The remaining degree is the scale of the translation. Given these 5DOF of multi-camera system motion (rotation and translation direction), we can compensate for the rotation of the system. Our approach is based on the observation that given the temporal epipolar geometry of one of the cameras, the position of the epipole in each of the other cameras of the multi-camera system is restricted to a line in the image. Hence, the scale as the remaining degree of freedom of the camera motion describes a linear subspace.

In the next section, we derive the mathematical basis of our approach to motion recovery.

3.3 Two Camera Systems – Theory

We consider a system involving two cameras, rigidly coupled with respect to each other. The cameras are assumed to be calibrated. Figure 3.3 shows the configuration of the two-camera system. The cameras are denoted by $\mathbf{C}_1$ and $\mathbf{C}_2$, at the starting position and $\mathbf{C}'_1$ and $\mathbf{C}'_2$ after a rigid motion.

We will consider the motion of the camera-pair to a new position. Our purpose is to determine the motion using image measurements. It is possible through standard techniques to compute the motion of the cameras up to scale, by determining the motion of just one of the cameras using point correspondences from that camera. However, from one camera, motion can be determined only up to scale. The direction of the camera translation may be

determined, but not the magnitude of the translation. It will be demonstrated in this chapter that a single correspondence from the second camera is sufficient to determine the scale of the motion, that is, the magnitude of the translation. This result is summarized in the following theorem.

Theorem 1. *Let a two camera system have initial configuration determined by camera matrices* $\mathtt{P}_1 = [\mathtt{I} \mid \mathbf{0}]$ *and* $\mathtt{P}_2 = [\mathtt{R}_2 \mid -\mathtt{R}_2\mathbf{C}_2]$. *Suppose it moves rigidly to a new position for which the first camera is specified by* $\mathtt{P}'_1 = [\mathtt{R}'_1 \mid -\lambda\mathtt{R}'_1\mathbf{C}'_1]$. *Then the scale of the translation,* λ*, is determined by a single point correspondence* $\mathbf{x}' \leftrightarrow \mathbf{x}$ *seen in the second camera according to the formula*

$$\mathbf{x}'^{\top}\mathtt{A}\mathbf{x} + \lambda\mathbf{x}'^{\top}\mathtt{B}\mathbf{x} = 0 \tag{3.1}$$

where $\mathtt{A} = \mathtt{R}_2\mathtt{R}'_1\,[(\mathtt{R}'_1{}^{\top} - \mathtt{I})\mathbf{C}_2]_\times\mathtt{R}_2{}^{\top}$ *and* $\mathtt{B} = \mathtt{R}_2\mathtt{R}'_1\,[\mathbf{C}'_1]_\times\mathtt{R}_2{}^{\top}$. *In this chapter* $[a]_\times b$ *denotes the skew-symmetric matrix inducing the cross product* $a \times b$.

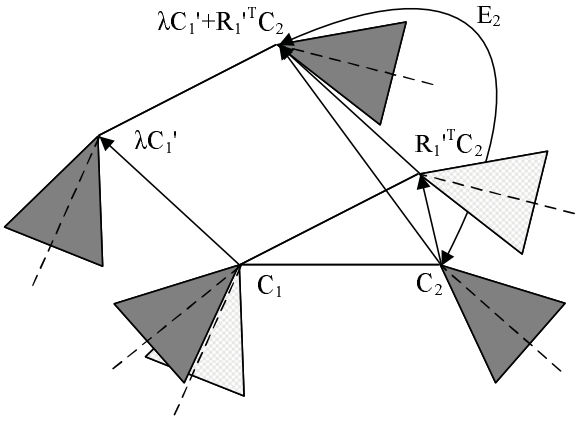

Figure 3.3: Motion of a multi-camera system consisting of two rigidly coupled conventional cameras.

In order to simplify the derivation we assume the coordinate system is centered on the initial position of the first camera, so that $\mathtt{P}_1 = [\mathtt{I} \mid \mathbf{0}]$. Any other coordinate system is easily transformed to this one by a Euclidean change of coordinates.

Observe also that after the motion, the first camera has moved to a new position with camera center at $\lambda \mathbf{C}_1'$. The scale is unknown at this point because in our method we propose as a first step determining the motion of the cameras by computing the essential matrix of the first camera over time. This allows us to compute the motion up to scale only. Thus, the scale λ remains unknown. We now proceed to derive Theorem 1. Our immediate goal is to determine the camera matrix for the second camera after the motion. First note that the camera $\mathtt{P}_1'$ may be written as

$$\mathtt{P}_1' = [\mathtt{I} \mid \mathbf{0}] \begin{bmatrix} \mathtt{R}_1' & -\lambda \mathtt{R}_1' \mathbf{C}_1' \\ \mathbf{0}^\top & 1 \end{bmatrix} = \mathtt{P}_1 \mathtt{T} \, .$$

where the matrix $\mathtt{T}$, is the Euclidean transformation induced by the motion of the camera pair. Since the second camera undergoes the same Euclidean motion, we can compute the camera $\mathtt{P}_2'$ to be

$$\begin{aligned} \mathtt{P}_2' &= \mathtt{P}_2 \mathtt{T} \\ &= [\mathtt{R}_2 \mid -\mathtt{R}_2 \mathbf{C}_2] \begin{bmatrix} \mathtt{R}_1' & -\lambda \mathtt{R}_1' \mathbf{C}_1' \\ \mathbf{0}^\top & 1 \end{bmatrix} \\ &= [\mathtt{R}_2 \mathtt{R}_1' \mid -\lambda \mathtt{R}_2 \mathtt{R}_1' \mathbf{C}_1' - \mathtt{R}_2 \mathbf{C}_2] \\ &= \mathtt{R}_2 \mathtt{R}_1' [\mathtt{I} \mid -(\lambda \mathbf{C}_1' + \mathtt{R}_1'^\top \mathbf{C}_2)] \, . \end{aligned} \tag{3.2}$$

From the form of the two camera matrices $\mathtt{P}_2$ and $\mathtt{P}_2'$, we may compute the essential matrix $\mathtt{E}_2$ for the second camera.

$$\begin{aligned} \mathtt{E}_2 &= \mathtt{R}_2 \mathtt{R}_1' [\lambda \mathbf{C}_1' + \mathtt{R}_1'^\top \mathbf{C}_2 - \mathbf{C}_2]_\times \mathtt{R}_2^\top \\ &= \mathtt{R}_2 \mathtt{R}_1' [\mathtt{R}_1'^\top \mathbf{C}_2 - \mathbf{C}_2]_\times \mathtt{R}_2^\top + \lambda \mathtt{R}_2 \mathtt{R}_1' [\mathbf{C}_1']_\times \mathtt{R}_2^\top \\ &= \mathtt{A} + \lambda \mathtt{B} \ . \end{aligned} \tag{3.3}$$

Now, given a single point correspondence $\mathbf{x}' \leftrightarrow \mathbf{x}$ as seen in the second camera, we may determine the value of λ, the scale of the camera translation. The essential matrix equation $\mathbf{x'}^\top \mathtt{E}_2 \mathbf{x} = 0$ yields $\mathbf{x'}^\top \mathtt{A}\mathbf{x} + \lambda \mathbf{x'}^\top \mathtt{B}\mathbf{x} = 0$, and hence:

$$\lambda = -\frac{\mathbf{x'}^\top \mathtt{A}\mathbf{x}}{\mathbf{x'}^\top \mathtt{B}\mathbf{x}} = -\frac{\mathbf{x'}^\top \left(\mathtt{R}_2 \mathtt{R}'_1 [\mathtt{R}'_1{}^\top \mathbf{C}_2 - \mathbf{C}_2]_\times \mathtt{R}_2{}^\top\right) \mathbf{x}}{\mathbf{x'}^\top \left(\mathtt{R}_2 \mathtt{R}'_1 [\mathbf{C}'_1]_\times \mathtt{R}_2{}^\top\right) \mathbf{x}} \tag{3.4}$$

.

So each correspondence in the second camera provides a measure for the scale. In the next section, we give a geometric interpretation for this constraint.

3.3.1 Geometric Interpretation

The situation may be understood via a different geometric interpretation, shown in Figure 3.4. We note from (3.2) that the second camera moves to a new position $\mathbf{C}'_2(\lambda) = \mathtt{R}'_1{}^\top \mathbf{C}_2 + \lambda \mathbf{C}'_1$. The locus of this point for varying values of λ is a straight line with its direction vector $\mathbf{C}'_1$, passing through the point $\mathtt{R}'_1{}^\top \mathbf{C}_2$. From its new position, the camera observes a point at position $\mathbf{x}'$ in its image plane. This image point corresponds to a ray $\mathbf{v}'$ along which the 3D point $\mathbf{X}$ must lie. If we think of the camera as moving along the line $\mathbf{C}'_2(\lambda)$ (the locus of possible final positions of the second camera center), then this ray traces out a plane Π; the 3D point $\mathbf{X}$ must lie on this plane.

On the other hand, the point $\mathbf{X}$ is also seen (as image point $\mathbf{x}$) from the initial position of the second camera, and hence lies along a ray $\mathbf{v}$ through $\mathbf{C}_2$. The point where this ray meets the plane Π must be the position of the point $\mathbf{X}$. In turn, this determines the scale factor λ.

3.3.2 Critical configurations

This geometric interpretation allows us to identify critical configurations in which the scale factor λ cannot be determined. As shown in Figure 3.4, the 3D point $\mathbf{X}$ is the intersection of

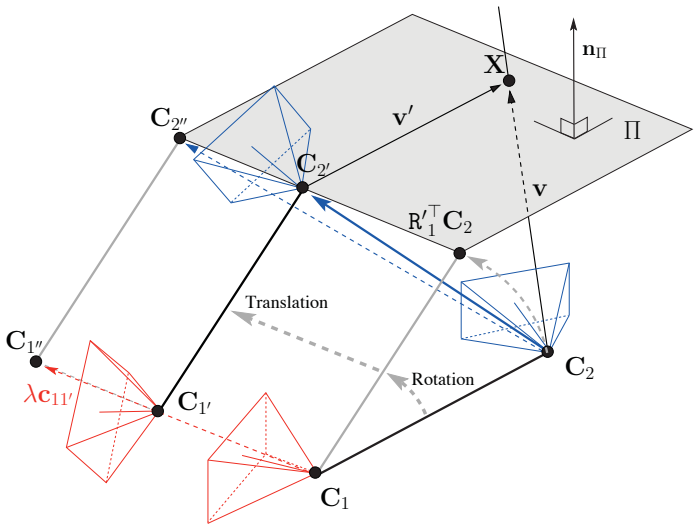

Figure 3.4: The 3D point $\mathbf{X}$ must lie on the plane traced out by the ray corresponding to $\mathbf{x}'$ for different values of the scale λ. It also lies on the ray corresponding to $\mathbf{x}$ through the initial camera center $\mathbf{C}_2$.

the plane Π with a ray $\mathbf{v}$ through the camera center $\mathbf{C}_2$. If the plane does not pass through $\mathbf{C}_2$, then the point $\mathbf{X}$ can be located as the intersection of plane and ray. Thus, a critical configuration can only occur when the plane Π passes through the second camera center, $\mathbf{C}_2$.

According to the construction, the line $\mathbf{C}'_2(\lambda)$ lies on the plane Π. For different 3D points $\mathbf{X}$, and corresponding image measurement $\mathbf{x}'$, the plane will vary, but always contain the line $\mathbf{C}'_2(\lambda)$. Thus, the planes Π corresponding to different points $\mathbf{X}$ form a pencil of planes hinged around the axis line $\mathbf{C}'_2(\lambda)$. Unless this line actually passes through $\mathbf{C}_2$, there will be at least one point $\mathbf{X}$ for which $\mathbf{C}_2$ does not lie on the plane Π, and this point can be used to determine the point $\mathbf{X}$, and hence the scale.

Finally, if the line $\mathbf{C}'_2(\lambda)$ passes through the point $\mathbf{C}_2$, then the method will fail. In this case, the ray corresponding to any point $\mathbf{X}$ will lie within the plane Π, and a unique point of intersection cannot be found.

In summary, if the line $\mathbf{C}'_2(\lambda)$ does not pass through the initial camera center $\mathbf{C}_2$, almost any point correspondence $\mathbf{x}' \leftrightarrow \mathbf{x}$ may be used to determine the point $\mathbf{X}$ and the translation

scale λ. The exceptions are point correspondences given by points $\mathbf{X}$ that lie in the plane defined by the camera center $\mathbf{C}_2$ and the line $\mathbf{C}'_2(\lambda)$ as well as far away points for which Π and $\mathbf{v}$ are almost parallel. The interested reader may wish to read another analysis of the critical configurations for scale estimation in non-overlapping multi-camera systems given in (Kim and Chung, 2006).

If on the other hand, the line $\mathbf{C}'_2(\lambda)$ passes through the center $\mathbf{C}_2$, then the method will always fail. It may be seen that this occurs most importantly if there is no camera rotation, namely $\mathtt{R}'_1 = \mathtt{I}$. In this case, we see that $\mathbf{C}'_2(\lambda) = \mathbf{C}_2 + \lambda\mathbf{C}'_1$, which passes through $\mathbf{C}_2$. It is easy to give an algebraic condition for this critical condition. Since $\mathbf{C}'_1$ is the direction vector of the line, the point $\mathbf{C}_2$ will lie on the line precisely when the vector $\mathtt{R}'^{\top}_1\mathbf{C}_2 - \mathbf{C}_2$ is in the direction $\mathbf{C}'_1$. This gives a condition for singularity $(\mathtt{R}'^{\top}_1\mathbf{C}_2 - \mathbf{C}_2) \times \mathbf{C}'_1 = \mathbf{0}$, or rearranging this expression, and observing that the vector $\mathbf{C}_2 \times \mathbf{C}'_1$ is perpendicular to the plane of the three camera centers $\mathbf{C}_2$, $\mathbf{C}'_1$ and $\mathbf{C}_1$ (the last of these being the coordinate origin), we may state:

Theorem 2. *The critical condition for singularity for scale determination is*

$$(\mathtt{R}'^{\top}_1\mathbf{C}_2) \times \mathbf{C}'_1 = \mathbf{C}_2 \times \mathbf{C}'_1 \ .$$

In particular, the motion is not critical unless the axis of rotation is perpendicular to the plane determined by the three camera centers $\mathbf{C}_2$*,* $\mathbf{C}'_1$ *and* $\mathbf{C}_1$*.*

Intuitively, critical motions occur when the rotation induced translation $\mathtt{R}'^{\top}_1\mathbf{C}_2 - \mathbf{C}_2$ is aligned with the translation $\mathbf{C}'_1$. The most common motion that causes a critical condition is when the camera system translates but has no rotation. Another common, but less obvious, critical motion occurs when both camera paths move along concentric circles. This configuration is illustrated in Figure 3.5. A vehicle borne multi-camera system turning at a constant rate undergoes critical motion, but not when it enters and exits a turn.

Detecting critical motions is important to determining when the scale estimates are

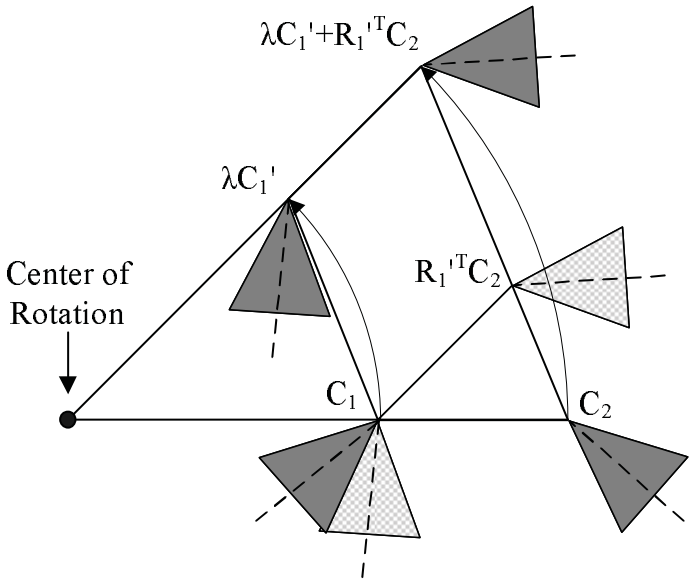

Figure 3.5: Critical motion due to constant rotation rate

reliable. One method to determine the criticality of a given motion is to use the approach of (Frahm and Pollefeys, 2006). We need to determine the dimension of the space that includes our estimate of the scale. To do this we double the scale λ and measure the difference in the fraction of inliers to the essential matrix of our initial estimate and the doubled scale essential matrix. If a large proportion of inliers are not lost when the scale is doubled then the scale is not observable from the data. If the scale is observable, the deviation from the estimated scale value would cause the correspondences to violate the epipolar constraint, which means they are outliers to the constraint for the doubled scale. When the scale is ambiguous, doubling the scale does not cause correspondences to be classified as outliers. This method proved to work on real data sets in practice.

3.4 Algorithm

Figure 3.6 shows an algorithm to solve relative motion of two generalized cameras from 6 rays with two centers where 5 rays meet one center and a sixth ray meets the other center. First, we use 5 correspondences in one ordinary camera to estimate an essential

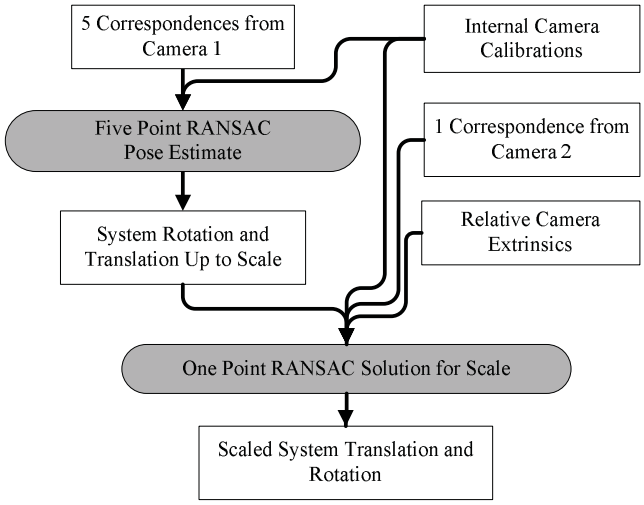

Figure 3.6: Algorithm for estimating 6DOF motion of a multi-camera system with non-overlapping fields of view.

matrix between two frames in time. The algorithm used to estimate the essential matrix from 5 correspondences is the method by Nistér (Nistér, 2003). It is also possible to use a simpler algorithm, which gives the same result developed by Li and Hartley (Li and Hartley, 2006). The 5 correspondences are selected by the RANSAC (Random Sample Consensus) algorithm (Bolles and Fischler, 1981). The distance between a selected feature and its corresponding epipolar line is used as an inlier criterion in the RANSAC algorithm. The essential matrix is decomposed into a skew-symmetric matrix of translation and a rotation matrix. When decomposing the essential matrix into rotation and translation the chirality constraint is used to determine the correct configuration (Hartley and Zisserman, 2004). At this point, the translation is recovered up to scale.

To find the scale of translation, we use Eq. 3.4 with RANSAC. One correspondence is randomly selected from the second camera and is used to calculate a scale value based on the constraint given in Eq. 3.4. We have also used a variant of the pbM-Estimator (Chen and Meer, 2003) to find the initial scale estimate with similar results and speed to the RANSAC approach. This approach forms a continuous function based on the discrete scale estimates

from each of the correspondences in the second camera and selects the maximum of that continuous function as the initial scale estimate.

Based on this scale factor, the translation direction and rotation of the first camera, and the known extrinsics between the cameras, an essential matrix is generated for the second camera. Inlier correspondences in the second camera are then determined based on their distance to the epipolar lines. A linear least squares calculation of the scale factor is then made with all of the inlier correspondences from the second camera. This linear solution is refined with a non-linear minimization technique using the graduated non-convexity (GNC) function (Blake and Zisserman, 1987) which takes into account the influence of all correspondences, not just the inliers of the RANSAC sample, in calculating the error. This error function measures the distance of all correspondences to their epipolar lines and smoothly varies between zero for perfect correspondence and one for an outlier with distance to the epipolar line greater than some threshold. One could just as easily take single pixel steps from the initial linear solution in the direction which maximizes inliers, or equivalently minimizing the robust error function. The non-linear minimization simply allows us to select step sizes depending on the sampled Jacobian of the error function, which should converge faster than single pixel steps and allows for sub-pixel precision.

Following refinement of the scale estimate, the inlier correspondences of the second camera are calculated and their number is used to score the current RANSAC solution. The final stage in the scale estimation algorithm is a bundle adjustment of the multi-camera system's motion. Inliers are calculated for both cameras and they are used in a bundle adjustment refining the rotation and scaled translation of the total, multi-camera system.

While this algorithm is described for a system consisting of two cameras, it is relatively simple to extend the algorithm to use any number of rigidly mounted cameras. The RANSAC for the initial scale estimate, initial linear solution and non-linear refinement are performed over correspondences from all cameras other than the camera used in the five-point pose estimate. The final bundle adjustment is then performed over all of the system's

cameras.

3.5 Experiments

We begin with results using synthetic data to show the algorithm's performance over varying levels of noise and different camera system motions. Following these results, we show the system operating on real data and measure its performance using data from a GPS/INS (inertial navigation system). The GPS/INS measurements are post-processed and are accurate to $4cm$ in position and 0.03 degrees in rotation, providing a good basis for error analysis.

3.5.1 Synthetic Data

We use results on synthetic data to demonstrate the performance of the 6DOF motion estimate in the presence of varying levels of normally distributed Gaussian noise on the correspondences over a variety of motions. A set of 3D points was generated within the walls of an axis-aligned cube. Each cube wall consisted of 5000 3D points randomly distributed within a $20m$ x $20m$ x $0.5m$ volume. The two-camera system, which has an inter-camera distance of $1.9m$, a 100^o angle between optical axes and non-overlapping fields of view, is initially positioned at the center of the cube, with identity rotation. A random motion for the camera system was then generated. The camera system's rotation was generated from a uniform $\pm 6^o$ distribution sampled independently in each Euler angle. Additionally, the system was translated by a uniformly distributed distance of $0.4m$ to $0.6m$ in a random direction. A check for degenerate motion is performed by measuring the distance between the epipole of the second camera (see Fig. 3.3) due to rotation of the camera system and the epipole due to the combination of rotation and translation. This check can be performed because we have perfect knowledge of the camera motion in synthetic data. Only results of non-degenerate motions with epipole separations equivalent to a 5^o angle between the

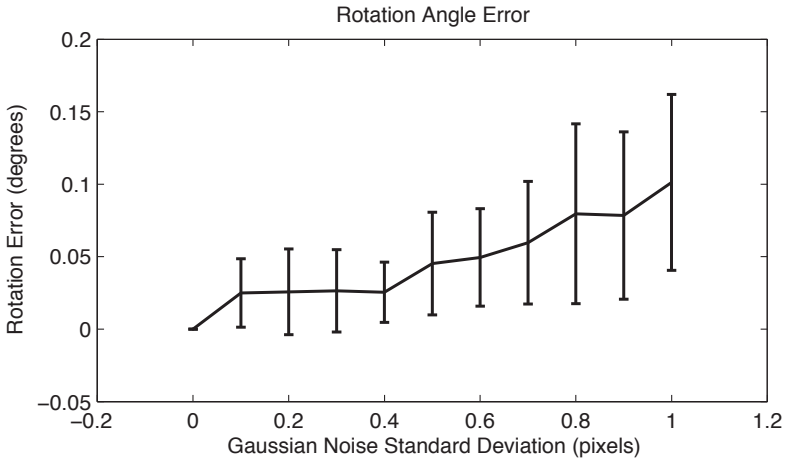

Figure 3.7: Angle Between True and Estimated Rotations, Synthetic Results of 100 Samples Using Two Cameras

translation vector and the rotation induced translation vector are shown. Results are given for 100 sample motions for each of the different values of normally distributed, zero mean Gaussian white noise added to the projections of the 3D points into the system's cameras. The synthetic cameras have calibration matrices and fields-of-view that match the cameras used in our real multi-camera system. Each real camera has an approximately 40^o x 30^o field-of-view and a resolution of 1024 x 768 pixels.

Results on synthetic data are shown in Figures 3.7 to 3.10. One can see that the system is able to estimate the rotation (Fig. 3.7) and translation direction (Fig. 3.8) well given noise levels that could be expected using a 2D feature tracker on real data. Figure 3.9 shows a plot of $\|T_{est} - T_{true}\| \,/\, \|T_{true}\|$. This ratio measures both the accuracy of the estimated translation direction, as well as the scale of the translation and would ideally have a value of zero because the true and estimated translation vectors would be the same. Given the challenges of translation estimation and the precise rotation estimation, we use this ratio as the primary performance metric for the 6DOF motion estimation algorithm. The translation vector ratio, along with the rotation error plot, demonstrate that the novel system performs well given a level of noise that could be expected in real tracking results.

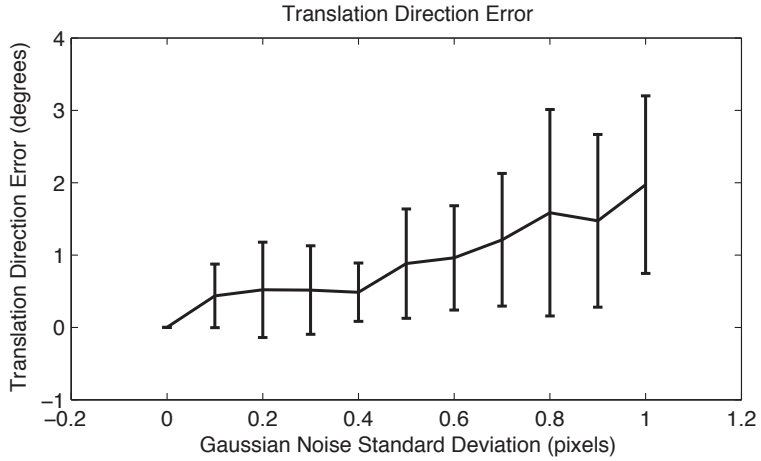

Figure 3.8: Angle Between True and Estimated Translation Vectors, Synthetic Results of 100 Samples Using Two Cameras

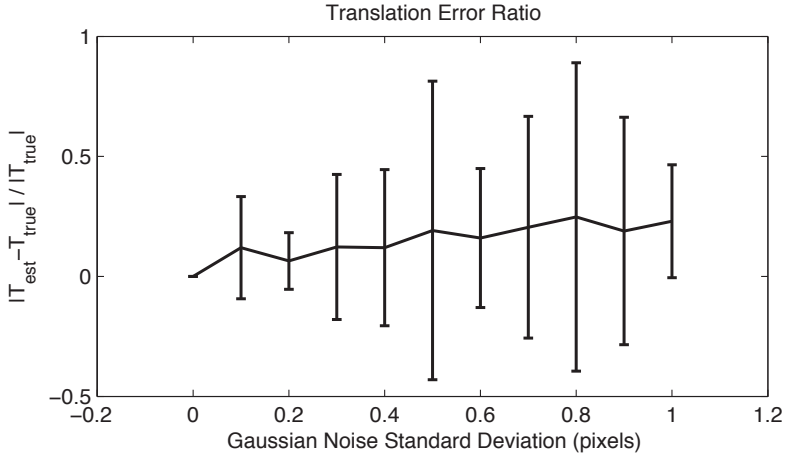

Figure 3.9: Scaled Translation Vector Error, Synthetic Results of 100 Samples Using Two Cameras

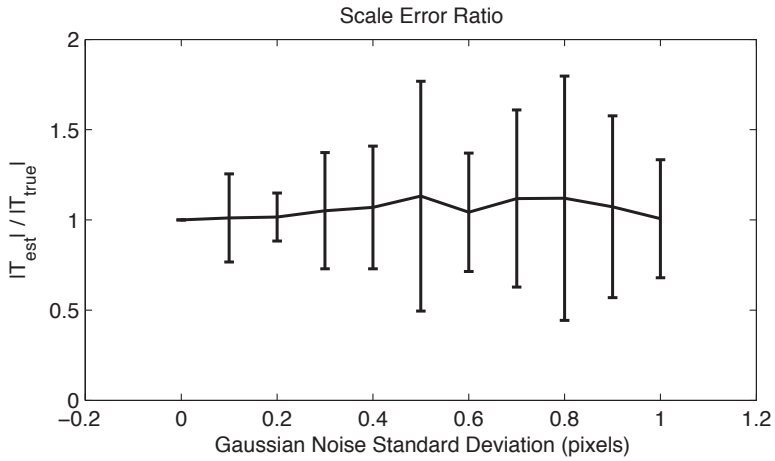

Figure 3.10: Scale Ratio, Synthetic Results of 100 Samples Using Two Cameras

3.5.2 Real data

For a performance analysis on real data, we collected video using an eight-camera system mounted on a vehicle. The system included a highly accurate GPS/INS unit, which allows comparisons of the scaled camera system motion calculated with our method to ground truth measurements. The eight cameras have almost no overlap to maximize the total field-of-view and are arranged in two clusters facing toward the opposite sides of the vehicle. In each cluster, the camera centers are within $25cm$ of each other. A camera cluster is shown in Fig. 3.1. The camera clusters are separated by approximately $1.9m$ and the line between the camera clusters is approximately parallel with the rear axle of the vehicle. Three of the four cameras in each cluster cover a horizontal field-of-view on each side of the vehicle of approximately 120^o x 30^o. A fourth camera points to the side of the vehicle and upward. Its principle axis has an angle of 30^o with the horizontal plane of the vehicle, which is colinear with the optical axes of the other three cameras.

In these results on real data, we take advantage of the fact that we have six horizontal cameras and use all of the cameras to calculate the 6DOF system motion. The upward facing cameras were not used because they only recorded sky in the sequence. For each pair of frames recorded at different times, each camera in turn is selected and the five-point pose

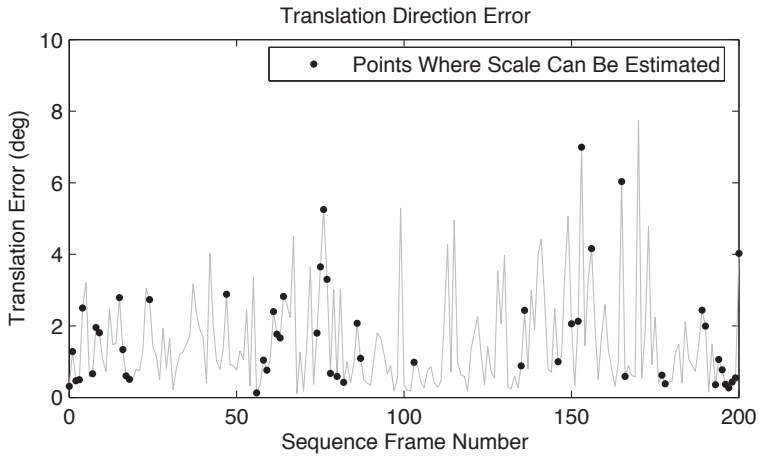

Figure 3.11: Angle Between True and Estimated Translation Vectors, Real Data with Six Cameras

estimate is performed for that camera using correspondences found using a KLT (Lucas and Kanade, 1981) 2D feature tracker. The other cameras are then used to calculate the scaled motion of the camera system using the five-point estimate from the selected camera as an initial estimate of the camera system rotation and translation direction. The 6DOF motion solution for each camera selected for the five-point estimate is scored according to the fraction of inliers of all other cameras. The motion with the largest fraction of inliers is selected as the 6DOF motion for the camera system.

In table 3.1, we show the effect of critical motions described in section 3.3.2 over a sequence of 200 frames. Critical motions were detected using the QDEGSAC (Frahm and Pollefeys, 2006) approach described in that section. Even with critical motion, the system degrades to the standard 5DOF motion estimation from a single camera and only the scale remains ambiguous as shown by the translation direction and rotation angle error in Figures 3.11 and 3.12. This graceful degradation to the one camera motion estimation solution means that the algorithm solves for all of the possible degrees of freedom of motion given the data provided to it.

In this particular experiment, the system appears to consistently underestimate the scale with our multi-camera system when the motion is non-critical. This is likely due to a

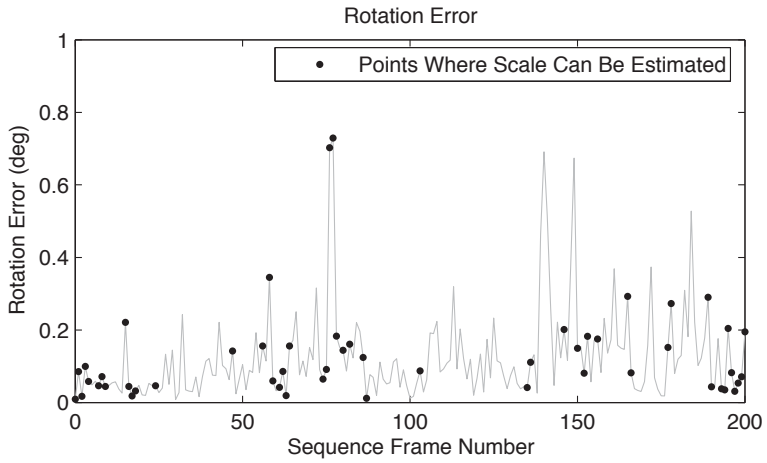

Figure 3.12: Angle Between True and Estimated Rotations, Real Data with Six Cameras

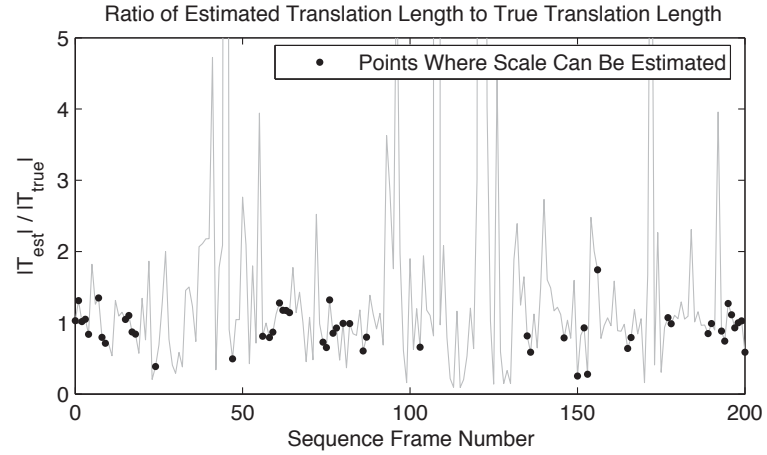

Figure 3.13: Scale Ratio, Real Data with Six Cameras

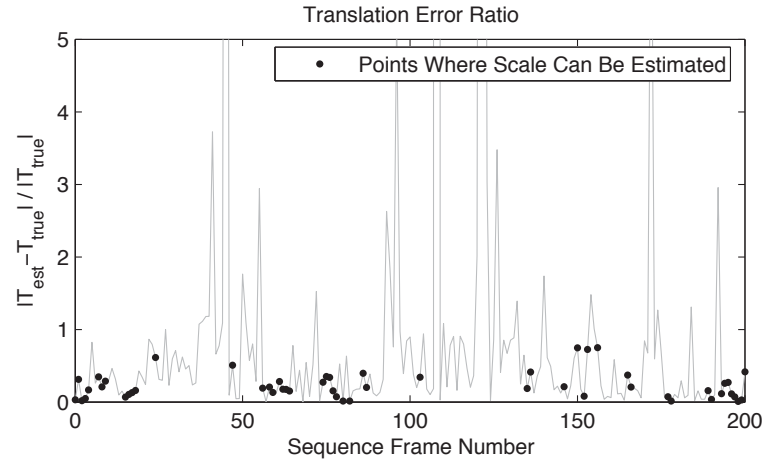

Figure 3.14: Scaled Translation Vector Difference from Ground Truth, Real Data with Six Cameras

$\frac{\|T_{est}-T_{true}\|}{\|T_{true}\|}$	0.23 ± 0.19
$\frac{\|T_{est}\|}{\|T_{true}\|}$	0.90 ± 0.28

Table 3.1: Relative translation vector error including angle and error of relative translation vector length $mean \pm std.dev$.

combination of error in the camera system extrinsics and error in the GPS/INS ground truth measurements.

Figure 3.15 shows the path of the vehicle mounted multi-camera system and locations where the scale can be estimated. From the map, it is clear that the scale cannot be estimated in straight segments as well as in smooth turns. This is due to the constant rotation rate critical motion condition described in section 3.3.2.

We selected a small section of the camera path circled in figure 3.15 and used a calibrated structure from motion (SfM) system similar to the system used in (Nistér, 2003) to reconstruct the motion of one of the system's cameras. For a ground truth measure of scale error accumulation, we scaled the distance traveled by a camera between two frames at the beginning of this reconstruction to match the true scale of the camera motion according to the GPS/INS measurements. Figure 3.16 shows how error in the scale accumulates over the 200 frames (recorded at 30 frames per second) of the reconstruction. We then processed the scale estimates from the 6DOF motion estimation system with a Kalman filter to determine the scale of the camera's motion over many frames and measured the error in the SfM reconstruction scale using only our algorithm's scale measurements. The scale drift estimates from the 6DOF motion estimation algorithm clearly measure the scale drift and provide a measure of absolute scale.

3.6 Conclusion

This chapter has introduced a novel algorithm that determines the 6DOF motion of a rigid multi-camera system with non-overlapping fields of view. We have provided a complete

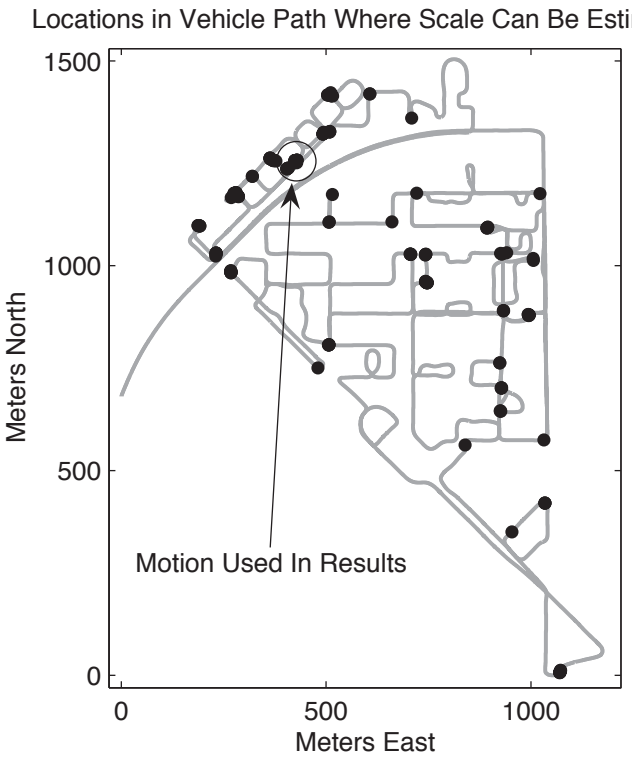

Figure 3.15: Map of vehicle motion showing points where scale can be estimated

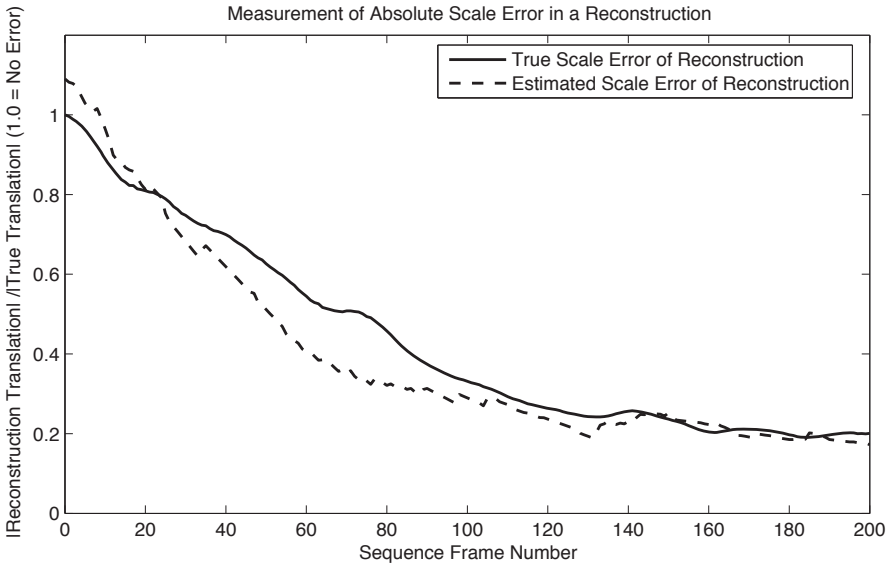

Figure 3.16: Scaled structure from motion reconstruction

analysis of the critical motions of the multi-camera system that make the absolute scale unobservable. Our algorithm can detect these critical motions and gracefully degrades to the estimation of the epipolar geometry. We have demonstrated the performance of our solution through both synthetic and real motion sequences. Additionally, we embedded our novel algorithm in a structure from motion system to demonstrate that our technique allows the determination absolute scale without requiring overlapping fields of view.

The next chapter will introduce a different minimal solution method for the six degree of freedom motion of a partially overlapping stereo camera. The approach is designed to overcome the degeneracies inherent estimating six degree of freedom motion for non-overlapping cameras by taking advantage of a small region of overlap in the rigidly mounted cameras' fields of view.

CHAPTER 4

Scaled Motion with a Slightly Overlapping Stereo Camera

4.1 Introduction

In this chapter, we present a new minimal solution method for use in stereo camera based structure from motion (SfM) or visual simultaneous localization and mapping (VSLAM). This novel minimal solution method overcomes the degeneracies in absolute scaled motion estimation inherent in non-overlapping rigid two camera systems, including the most common case of pure translational motion.

Our principal application is VSLAM for a humanoid robot. The approach proposed in this work is analogous to human vision where both eyes overlap in only part of the total viewing frustum. Excluding prior models humans possess, such as relative sizes of objects, expected relative positions, expected ego-motion etc, depth could be perceived from the region of overlap between our eyes while rotation is derived from both overlapping and non-overlapping regions. This configuration of eyes (or cameras) provides a large total field-of-view for the combined camera system while at the same time allowing the scale to be fixed based on triangulated features in the camera's region of overlap. This gives the best of what a two-camera system can deliver, a wide field-of-view for accurate rotation estimation with an absolutely scaled translation measurement.

Our solution method is based on the observation that a feature visible in all four cameras

(corresponding to a pair of cameras at two poses) constrains the relative pose of the second stereo camera to be on a sphere around this particular feature, which has a known position relative to the first stereo camera pose from triangulation, as shown in Figure 4.1. Features seen in only the left or right camera at both poses (two-view features) are labeled $S_{1..3}$, and the feature seen in all four cameras is labeled Q (for quad or four-view feature). The constraint imposed by the four-view feature leaves three degrees of freedom: two degrees for the location of the second camera on the induced sphere, and one for the rotation in the tangent plane to the sphere.

It appears natural to employ two-view correspondences visible only either in the left or right view. There is a surprising degeneracy in this setting, if e.g. the left camera of the stereo pair has the same distance from the triangulated four-view feature in both poses (which can be readily verified e.g. using Macaulay 2 (Grayson and Stillman). More importantly, this degeneracy also has an impact on the numerical accuracy of the returned solution, if the camera-point distances at both time instances do not vary significantly (which is usually the case in VSLAM settings). To avoid this degeneracy and to obtain well-conditioned solutions even in the small motion case, we use a pair of two-view correspondences in the left (or right) camera and one two-view correspondence in the other camera to solve for the remaining three degrees of freedom.

The major advantage of our approach is that it uses the total field-of-view of the stereo camera system to determine the rotation, which maximizes accuracy. Given a stereo pair with a small overlap between its views, one could triangulate 3D points in the region of overlap in the first stereo camera. And, use the three-point perspective pose solution (Haralick et al., 1994) or its generalized incarnation (Nistér, 2004) to find the relative pose of the second stereo camera with respect to the first camera. Due to the small overlap, the relative rotation and translation from these methods could be inaccurate because most of the 3D points would be in approximately the same viewing direction relative to the first camera. Our method fixes the rotation with features which do not have to be in the region

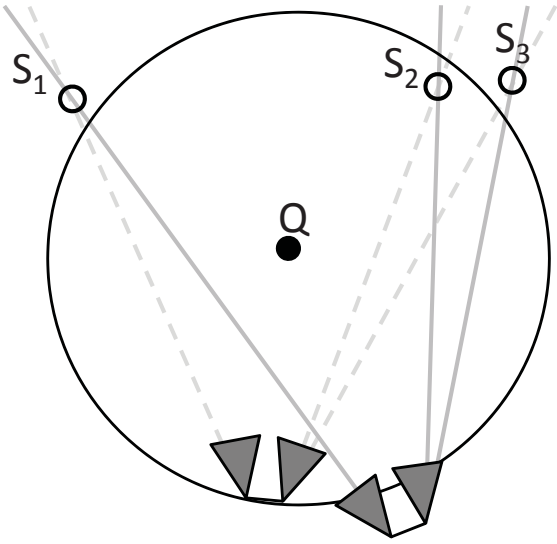

Figure 4.1: Geometry of stereo pair relative pose problem. Features seen in only the left or right camera are labeled S_i (two-view features). The feature seen in both left and right cameras at both times is labeled Q (four-view feature).

of overlap. With this method we get a more accurate rotation with small overlap, which in turn feeds into a more accurate translation measurement.

Our solver necessitates some modification to standard RANSAC (Bolles and Fischler, 1981) to account for the differing amount of information in a two-view correspondence and a twice-triangulated four-view correspondence. We develop a method to weigh the relative information in each type of inlier to determine the best solution in the RANSAC process. We also describe how to modify the RANSAC stopping criteria to account for two classes of measurement.

4.2 Background

Estimating SfM using multi-camera systems has been a topic of recent interest in the literature. In some instances, one can take advantage of a reduced degree of freedom (DOF) motion model to simplify SfM. This can be used to estimate the planar motion of a wheeled robot for example. This approach was taken by Stewénius and Åström in (Stewénius and

Åström, 2004) where they theoretically derive solutions for planar SfM for many different combinations of rigidly mounted cameras, features, and multi-camera system poses. Scaramuzza et al. (Scaramuzza et al., 2009) take advantage of the fact that a vehicle moves with planar non-holonomic motion to calculate the absolutely scaled structure and motion using only one point correspondence. In contrast, our approach is to be used with stereo camera systems allowed to move in all 6DOF.

A few approaches to solve for six degree of freedom motion of a multi-camera system exist which can solve for the system's motion without any overlapping views. Kim et al. described one such method in (Kim et al., 2007). They first calculate essential matrices for all the system's cameras and decompose these essential matrices into rotation and translation direction. They show that the scale of the motion can then be solved as a triangulation problem.

Another approach was presented in Chapter 3 and in Clipp et al. (2008) where the same problem is addressed using a minimal number of six feature correspondences: five from one camera are used to determine the rotation and translation direction, and the sixth one from a different camera determines the translation scale. Both methods (Clipp et al., 2008; Kim et al., 2007) are based on decomposing the translation of the individual cameras into translation of the total system and the rotation induced translation. This can only be done if these two vectors are not parallel or close to parallel. Accordingly, these methods are only applicable to certain key-frames in any general video sequence.

The generalized epipolar constraint of multi-camera systems was developed by Pless in (Pless, 2003) to allow a network of rigidly-coupled cameras to be treated as a single imaging device for SfM. Pless suggests a linear 17-point solution for relative motion using the generalized epipolar constraint but left numerical results for this approach to future work. Li et al. show in (Li et al., 2008) that this standard linear 17-point approach to solve for relative motion using the generalized epipolar constraint based on singular value decomposition does not work in many common scenarios and propose a non-minimal, linear

solution that does.

In (Stewénius et al., 2005) Stewénius and Nistér show that there are two minimal cases for the relative pose of a generalized camera and develop a solution method for the relative pose of stereo cameras from six ray correspondences. They demonstrate that up to 64 solutions may exist for the proposed constraints. Their solution is degenerate if all cameras in the multi-camera system are on a line, which of course is the case with a stereo camera (our target here).

At the other end of the spectrum are methods that require all features used in pose estimation to be in the multi-camera system's region of overlap in the first pose. These include the before mentioned three-point perspective pose problem (Haralick et al., 1994) and the generalized three-point perspective pose problem (Nistér, 2004). While these methods are not vulnerable to the pure translation degeneracy, in contrast to (Clipp et al., 2008; Kim et al., 2007), they require sufficient overlap between the cameras' fields-of-view, necessitating a large trade-off between overlap and total field-of-view.

Our approach, using one four-view feature to fix the camera translation and three two-view correspondences to find the rotation, occupies a middle ground in the space of problems considered until now.

4.3 Solution Method

In this section, we describe our approach for relative pose estimation between two poses of a stereo rig. Let P_0 and P_1 denote the projection matrices of the left and the right camera for the first time instance, and P_0' and P_1' are those for the second time instance. Without loss of generality, we assume a rectified stereo system, i.e. $P_0 = (I|0)$ and $P_1 = (I|b)$, where $-b$ is the baseline between the cameras on the stereo rig. General configurations can be reduced to this case by appropriate rotation of the 2D feature positions (corresponding to 3D camera rays emerging from the projection center). We also assume the camera intrinsics and lens

distortion parameters are known, and that feature positions in the image are undistorted and normalized according to the intrinsic parameters.

Let $(R|t)$ denote the Euclidean transformation between the time instances, i.e.

$$P'_0 = (R|t) \quad \text{and} \quad P'_1 = (R|t+b).$$

The 3D point visible in both cameras has coordinates x for the first time instance and y in the second instance (always with respect to the left camera in the rig). The 3D points x and y are found through triangulating their associated correspondences in each stereo pair. Hence,

$$y = Rx + t,$$

and $t = y - Rx$. 2D feature matches $p_0 \leftrightarrow q_0$ visible only in the left camera must satisfy the epipolar constraint,

$$q_0^T E_0 p_0 = q_0^T [t]_\times R p_0 = 0. \tag{4.1}$$

The epipolar constraint for feature correspondences $p_1 \leftrightarrow q_1$ only visible in the right camera can be easily derived as

$$\begin{aligned} q_1^T E_1 p_1 &= q_1^T [b + t - Rb]_\times R p_1 \\ &= q_1^T [b + y - Rx - Rb]_\times R p_1 \;\; \text{with } [t = y - Rx] \\ &= q_1^T \big([b+y]_\times R - R[x+b]_\times \big) p_1, \end{aligned} \tag{4.2}$$

where we used the fact that $[Rx]_\times Ry = (Rx) \times (Ry) = R(x \times y) = R[x]_\times y$ for rotation matrices R.

Overall, Equations 4.1 and 4.2 allow us to express both essential matrices in terms of

the unknown rotation R, i.e.

$$E_0 = [t]_\times R = [y]_\times R - R[x]_\times, \text{ and} \tag{4.3}$$

$$E_1 = [b + y]_\times R - R[x + b]_\times. \tag{4.4}$$

In general, with two correspondences in the left camera and one correspondence in the right camera, there are three equations for the three degrees of freedom of the rotation. Using e.g. unit quaternions to represent the rotation matrix R, a polynomial system of four equations can be formulated, which contains the three epipolar constraints (two constraint equations for the two-view correspondence in the left camera and one constraint equation for the two-view correspondence in the right camera, or one left and two right)and the unit quaternion constraint. By computing the elimination ideal (e.g. (Cox et al., 1997)), a 16th degree univariate polynomial (with only even powers) is obtained. The two solutions differing only in sign correspond to the quaternions (q_0, q_1, q_2, q_3) and $(-q_0, -q_1, -q_2, -q_3)$, which actually represent the same rotation.

In our initial experiments we compute a Gröbner basis trace for the polynomial system of equations described above (using exact arithmetic in finite prime fields, as suggested in (Stewénius, 2005)), and generate efficient code automatically using Buchberger's algorithm to solve real instances of the pose estimation problem. In order to allow real-time processing, we utilize a root finding procedure based on Sturm bracketing instead of using one of the numerically more stable, but substantially slower approaches (e.g. (Bujnak et al., 2008; Byröd et al., 2007)). The observed numerical accuracy of this method degrades with decreasing baselines in terms of the 3D scene depths, which reduces its utility in real-world situations. Nevertheless, the assumption of small motion (in particular small rotations) of the camera system over time (also commonly employed in differential feature tracking methods) allows us to simplify the polynomial system as follows.

First, we represent the rotation R by modified Rodrigues parameters σ (Schaub and

Junkins, 2003),

$$R(\sigma) = I + \frac{8[\sigma]_\times^2 - 4(1 - \|\sigma\|^2)[\sigma]_\times}{(1 + \|\sigma\|^2)^2}, \tag{4.5}$$

where σ is a 3-vector. Since the modified Rodrigues parameters can be expressed in terms of the Euler axis $\bar{a}$ and angle θ as $\sigma = \bar{a}\tan(\theta/4)$, the linearization error increases with $\theta/4$ instead of e.g. $\theta/2$ for the classical Rodriguez parametrization. Hence, this particular representation relaxes the assumption of small rotation angles in comparison with other representations.

Under the assumption of small rotations, we approximate $R(\sigma)$ by its second order Taylor approximation and insert the resulting expression into Eqs. 4.3 and 4.4. We chose to use a second order Taylor series approximation so that our solution would work with larger rotations than a first order approximation would allow. The resulting polynomial system has three equations of degree two. The corresponding Groebner basis trace leads to an 8th degree polynomial and consists of only a few steps, hence the induced solution procedure is considered to be numerically stable. Root finding and back-substitution give the modified Rodrigues parameters σ and the corresponding rotation through Eq. 4.5, which is only an approximate solution to the original problem. The reported possible rotations are nonlinearly refined to satisfy Eqs. 4.3 and 4.4 precisely.

4.4 RANSAC Considerations

The minimal solution method described in the previous section uses two modes of data points—point feature matches in 3D space, and feature correspondences in 2D. Hence, we have to deviate from the uniform treatment of samples employed in traditional RANSAC settings.

The first modification addresses the refined stopping criterion to account for the potentially different inlier ratios for the two different types of correspondences. The algorithm

maintains the inlier ratio ϵ_l for feature correspondences visible in the left camera, the inlier ratio ϵ_r for features visible in the right camera, and the inlier ratio ϵ_d for features visible in both cameras (four-view features). Without loss of generality we assume that two temporal correspondences from the left camera and one correspondence from the right camera are utilized in the minimal solver. Then the modified stopping criterion is given by

$$n = \frac{\log(1-\alpha)}{\log(1-\epsilon_l^2\epsilon_r\epsilon_d)}, \tag{4.6}$$

where n is the number of samples required to achieve confidence α in the solution. Clearly, the term $\epsilon_l^2\epsilon_r\epsilon_d$ is the probability of selecting only inliers as required by our solution method (Section 4.3).

The inlier scoring function for a pose hypothesis also needs adjustment. In standard RANSAC the total count of inliers is used to score the current hypothesis, which assumes that all data points contain the same amount of information. In the given setup we face a mixed set of data points consisting of 3D points visible in 4 images in total, and 2D feature correspondences. In both cases the latent variables have 3 degrees of freedom per sample—the coordinates of the underlying 3D point. However, the dimension of the observed measurement is either 4-dimensional (two 2D feature positions in either the left or the right camera) or 8-dimensional (2D positions in both cameras at both points in time). Consequently, the residual error lies in the space orthogonal to the fitted manifold (Torr, 2002). Accordingly, the error space for 3D points visible in all cameras is 5-dimensional (8-3), and the residuals for points visible either in the left or right camera is 1-dimensional (4-3). Therefore, inlier features visible in four views carry five times more information than two view correspondences. Thus, the utilized weighting to combine the respective inlier

counts is given by

$$score = 5 \times \#Inliers(3D) + \#Inliers(left) + \#Inliers(right), \quad (4.7)$$

where $\#Inliers(3D)$ denotes the number of inliers among fully visible 3D points (four-view features), and $\#Inliers(left)$ and $\#Inliers(right)$ designate the inlier counts of the respective two-view correspondences.

4.5 Minimal Sets of Correspondences for Stereo Cameras

Our method uses a novel combination of image correspondences to determine the relative pose of a stereo camera. This leads to the question of what other combinations of correspondences can be used to solve for a stereo camera's relative pose. Four images are available as input to compute the relative pose between stereo cameras. Thus, a point feature may be detected and matched in all four (4-V), three (3-V) or only two views (2-V). Table 4.5 shows all of the combinations of respective feature correspondences, which result in a minimal set of constraints fully determining the relative pose of a stereo camera. Only three sets of constraints (out of 6 possible ones) have been used for relative pose estimation for stereo cameras in the literature so far.

The displayed minimal sets of correspondences are established by looking at independent constraints induced by a combination of correspondences. A four-view correspondence constrains 3DOF between the two stereo cameras, a three-view correspondence 2DOF and a one-view correspondence 1DOF. It is clear the total number of constraints generated by the correspondences needs to be exactly six for a minimal case. However, one must be careful to consider the redundancy in combinations of correspondences. A simple counting argument would suggest a relative pose with two four-view correspondences

Description	4-V	3-V	2-V
Figure 4.3	1	1	1
Chapter 4, Figure 4.1	1	0	3
P3P (Haralick et al., 1994; Nistér, 2004)	0	3	0
Figure 4.4	0	1	4
Figures 4.5 and 4.6	0	2	2
Chapter 3, Figure 3.3	0	0	6

Table 4.1: Combinations of N-view correspondences that minimally constrain the 6DOF relative pose of a stereo camera

and no others would be minimal. In fact this situation is under-determined because of its geometry.

Figures 4.3 to 4.6 illustrate additional minimal geometric configurations that could give rise to minimal solution methods. None of these methods has been verified by developing an algebraic solution method based on the geometry. These cases are described by their number of four-view, three-view and two-view correspondences in the following format case$< \#four-view, \#three-view, \#two-view >$. For example, the solution method using a non-overlapping stereo camera described in Chapter 3 is denoted case$< 0, 0, 6 >$. In the figures solid, thick lines connect camera centers with rays to points where the ray direction and point depth are both known. Dashed lines connect camera centers to points with rays where only the ray direction is known. Since the second camera in the stereo pair is only used to find the depth of three-view and four-view features, it is not drawn in the figures.

Case$< 1, 1, 1 >$ is shown in Figure 4.3. This geometry clearly fully constrains the two cameras since the four-view and three-view features constrain the two cameras so that their only degree of freedom is rotation about the vertical line through the two features. A single additional two-view correspondence should provide the additional constraint required to fully constrain the relative pose of the two cameras, P_0 and P_1.

Figure 4.4 illustrates case$< 0, 1, 4 >$ which also clearly reflects a fully constrained geometry. This can be seen by considering the solution method that might be applied to find

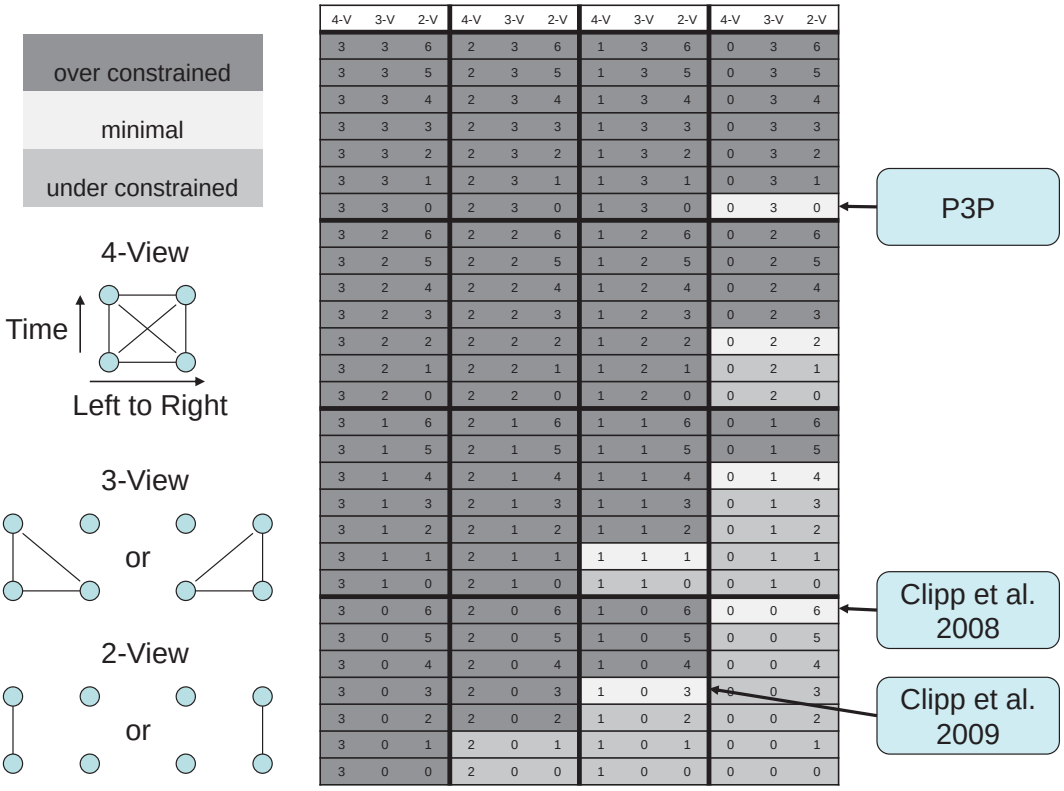

Figure 4.2: Over-constrained, minimally-constrained and under-constrained combinations of features for six degree of freedom motion estimation for a rigid, calibrated two-camera system.

the scaled transformation from P_0 to P_1. The five-point relative pose method Nistér, 2003 can be used with the five correspondences, ignoring for the moment that one of the correspondences has a known depth. The five-point method finds the transformation's rotation and translation direction. At this point the transformation is known up to an unknown scale factor. Using the three-view correspondence's depth this scale factor can be determined to recover the six degree of freedom, absolutely scaled transformation from P_0 to P_1. Additional combinations of three-view and two-view features may lead to other geometries for case$< 0, 1, 4 >$. In these geometries rather than all of the two-view features coming from the same camera, e.g. the left camera in the stereo pair, they might come from some combination of left to left and right to right camera correspondences. These combinations of correspondences might be looked at in future work.

Two possible geometries exist for case$< 0, 2, 2 >$ which we will refer to as case$< 0, 2, 2 >_a$ and case$< 0, 2, 2 >_b$. Case$< 0, 2, 2 >_a$ fully constrains the relative pose of P_0 and P_1. After the two three-view correspondences are included in the geometry, camera P_1 has two remaining degrees of freedom. It can rotate about the line through the two three-view features and it can move along the circle shown in Figure 4.5. These two degrees of freedom should be resolved with the two two-view correspondences. Upon inspection case$< 0, 2, 2 >_b$, Figure 4.6, does not appear to fully constrain the relative pose but it is included here because it warrants additional study.

4.6 Degenerate Cases

In this section, we describe certain configurations of features that lead to degeneracies in our solution for case$< 1, 0, 3 >$. The major reason we use two two-view correspondences in the left (or right) camera and one two-view correspondence in the other camera to solve for the rotation is a degeneracy that can occur when using correspondences only from one of the cameras. If all three two-view correspondences are selected from either the left or

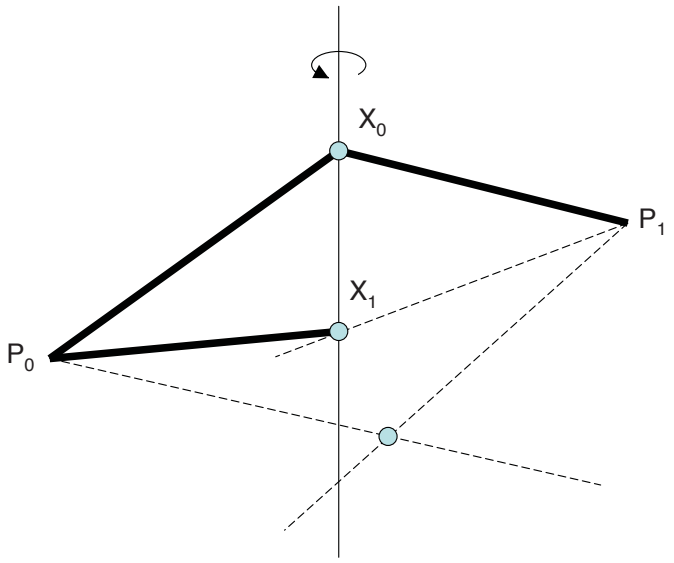

Figure 4.3: The minimal geometry consisting of one four-view, one three-view and one two-view correspondence, case$< 1, 1, 1 >$.

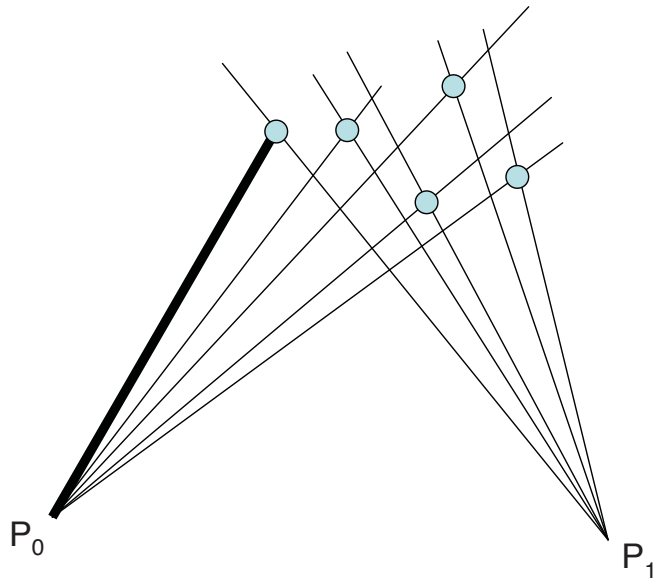

Figure 4.4: The minimal geometry consisting of one three-view and four two-view correspondences, case$< 0, 1, 4 >$.

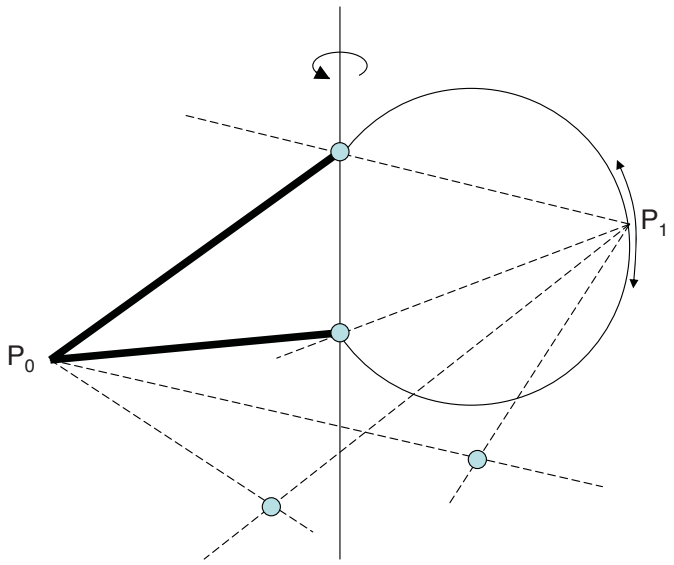

Figure 4.5: The first case of the minimal geometry consisting of two three-view and two two-view correspondences, case$< 0, 2, 2 >_a$. While an algebraic solution is not provided in this dissertation, this geometry appears to fully constrain the relative camera poses.

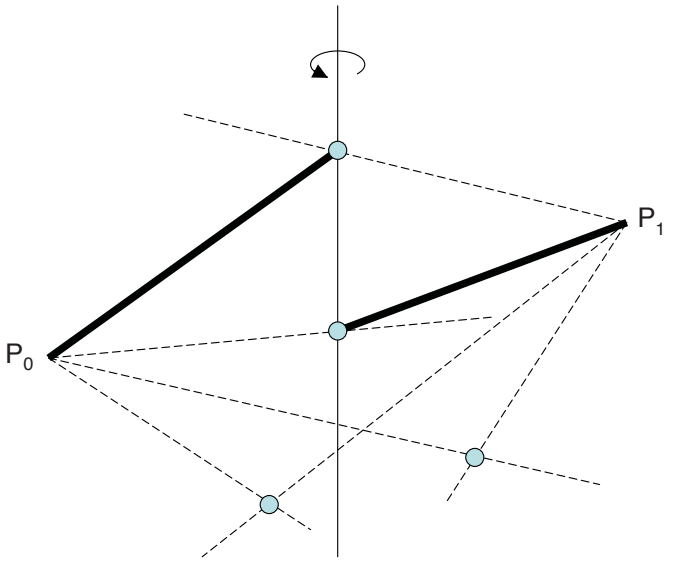

Figure 4.6: The second case of the minimal geometry consisting of two three-view and two two-view correspondences,case$< 0, 2, 2 >_b$. It is not immediately clear that this geometry fully constrains the relative camera poses but it may.

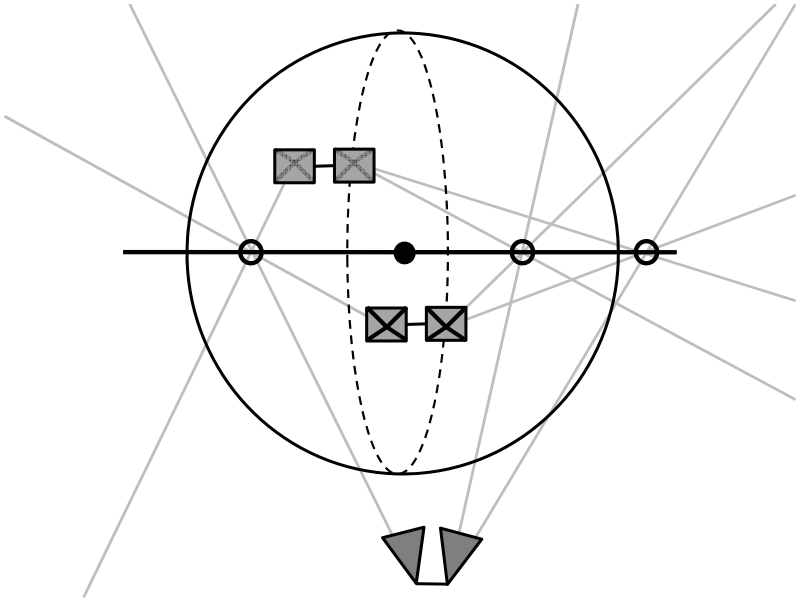

Figure 4.7: The first degenerate case. When all the features (two-view and four-view) lie on a 3D line through the four-view feature the second camera can be anywhere on a circle which is the intersection of the sphere and a plane orthogonal to the line containing the four-view feature.

the right camera to solve for the rotation, and the four-view feature is equidistant from the left camera at both poses, then this setting is degenerate and the 1D variety of solutions exists. This situation is resolved by utilizing two-view correspondences from both the left and right cameras to solve for the rotation.

One truly degenerate case that may arise with our method in practice is when all three features, which give rise to the 2D correspondences, lie on a line through the 3D point used in the minimal solution at the center of the sphere. In this case, the camera can be anywhere on a circle described by the intersection of the sphere and a plane through the center of the sphere orthogonal to the line. This configuration is depicted in Figure 4.7. This configuration might occur in man-made environments where straight lines are present. However, this configuration is a common degeneracy for all relative pose solution methods and can be easily avoided by never selecting correspondences that are all co-linear in both images.

4.7 Synthetic Experiments

In this section, we evaluate the performance of our minimal solver using two synthetic experiments. First, we evaluate the performance after nonlinear refinement of the solver under varying levels of image noise with and without outliers to test the solver's ability to deal with corrupted data. Second, we compare our solver to the three-point perspective pose solution without refinement while decreasing the overlap of the stereo pair by rotating the cameras on the rigid system around their vertical axes. In these and the experiments in Section4.8, the solution method based on the small angle approximation is used as it considerably simplifies the polynomial equations that must be solved to find the camera system motion.

The first experimental setup tests random motions of a stereo camera. For ease of explanation, we assume all units of length are in meters. The two cameras have a baseline of 0.5 m, have parallel optical axes, and are placed in a standard stereo camera configuration where both camera centers are on a line from the left optical axis to the right optical axis orthogonal to the axes. The first camera is placed with identity rotation and the left camera of the stereo head at the origin. Three dimensional feature points are distributed in a 20x20x20m volume in front of the camera. The second camera pose is generated by first randomly translating the camera between 0.2 and 3.5 meters in a random direction. The minimum translation reduces the effect of being close to the degenerate case when the two cameras are the same distance from the 3D feature, which is the center of rotation. The second stereo pair is then rotated randomly up to five degrees in each of the three rotation axes. Based on visibility we divide the 3D features into three classes: those that can be seen in both cameras of the stereo system at both times (four-view features), those that can be seen only in the left camera at both times, and those that can be seen only in the right. In this way we model the effect of a limited overlap in the fields of view.

We test the performance of our proposed minimal solver under varying levels of noise

and outliers. We use RANSAC (Bolles and Fischler, 1981) to find an initial solution and do a bundle adjustment on the inliers to refine the relative pose solution. Figure 4.8 shows the rotation and translation direction error with varying levels of noise added to the features in the images with and without outliers. In all Figures in this chapter, the rotation error is defined as the angle component of $R_{true}R_{est}^T$ in an axis-angle representation, and the translation error is the angle between the true and estimated translation vectors. We use the translation direction rather than distance because in situations where the features are relatively far from the camera compared to the stereo camera baseline the magnitude of the translation will be only weakly observable. By comparing the translation directions we compare quantities which should have greater observability in the data. Given that our method uses the 3D location of one feature, we triangulate the noisy image measurements of this feature in both stereo pairs independently and use the triangulated point locations as input to our solver. Image noise is reported in degrees. Note that 0.1° corresponds to two pixels for a camera with a sixty-degree field-of-view and 1200 pixels horizontal resolution. Figures 4.8 and 4.9 clearly show our solver is able to separate inliers from outliers in the presence of noise.

The second experiment is designed to test our method's performance vs. the three-point perspective pose method (P3P) in a typical indoor scenario. The camera is placed in a corridor, which has 3D features randomly distributed in the 0.1m thick walls. The corridor is 20m long, 2m high and 2m wide. The first stereo camera is placed in the middle of the corridor pointing down the corridor at the far wall, which is 10m away from the camera. The second stereo camera is randomly translated and rotated in a way that it is moving down the hall and points on the far wall are visible.

We progressively reduce the overlap between the cameras by rotating the left and right cameras' optical axes away from each other. We compare the accuracy of the relative pose calculated using our method and (P3P) after RANSAC but without non-linear refinement. This provides a measure of how close the RANSAC solution is to the true solution. The

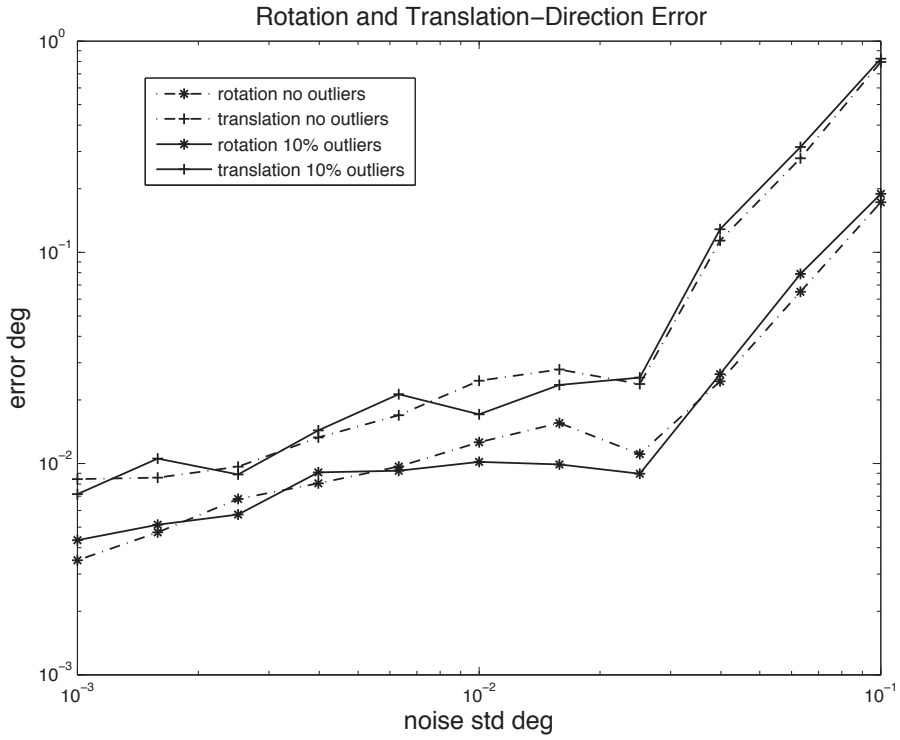

Figure 4.8: Absolute rotation error and translation direction error after non-linear refinement under varying noise with and without outliers. The slightly larger error with outliers at larger noise values may be due to the fact that fewer inlier features are used to calculate these solutions than the pure inlier sets. For example, if 100 features were used in the experiment then with 10% outliers only 90 features would be used to calculate the motion vs. 100 with 0% outliers.

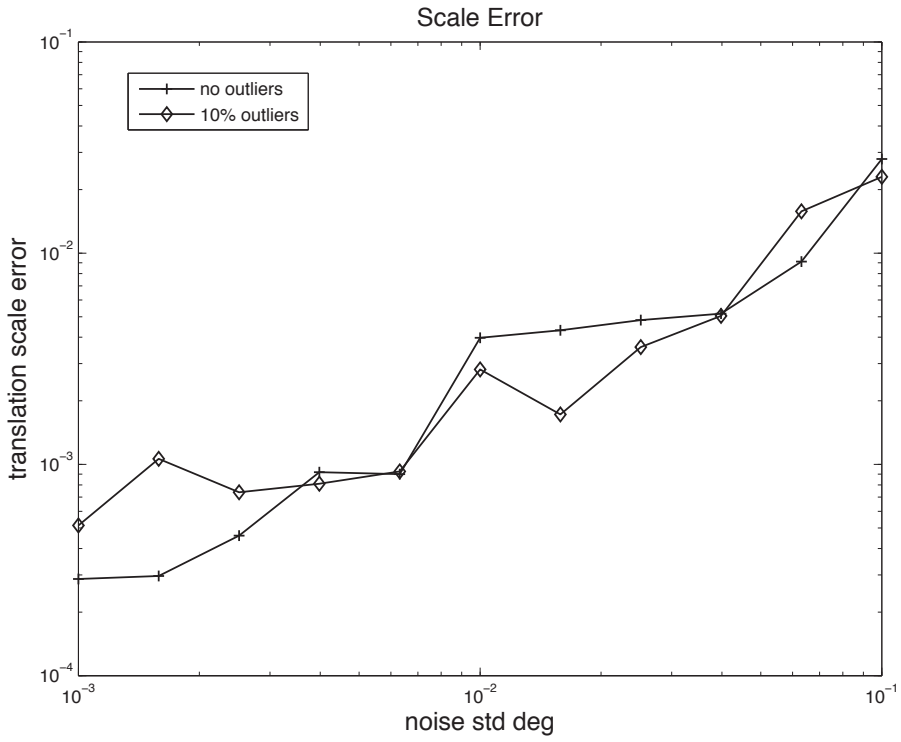

Figure 4.9: Error in the scale of camera translation under varying noise with and without outliers. The displayed error is given as $\big|\|T_{est}\| - \|T_{true}\|\big|/\|T_{true}\|$.

closer to the true solution the more likely it is that a non-linear refinement will find the global minimal error solution. We test both methods on exactly the same random motions, 3D feature locations and same noisy feature measurements over 1000 trials and report the average results.

For the P3P method we first triangulate 3D points between the left and right cameras of the stereo system at the initial pose, P_0. We then use the projections of these triangulated features in the left image of the stereo head at the second pose, P_0', to calculate the relative pose. We calculate inliers and outliers and score both the P3P solution and our solution method in the same manner. We use an adaptive stopping criterion so that we can compare the required number of RANSAC samples to reach 99% confidence. We also compare the rotation and translation direction error and scale error of the two methods. In Figures 4.10 through 4.12, the percentage on the legend shows the percent overlap at infinity between

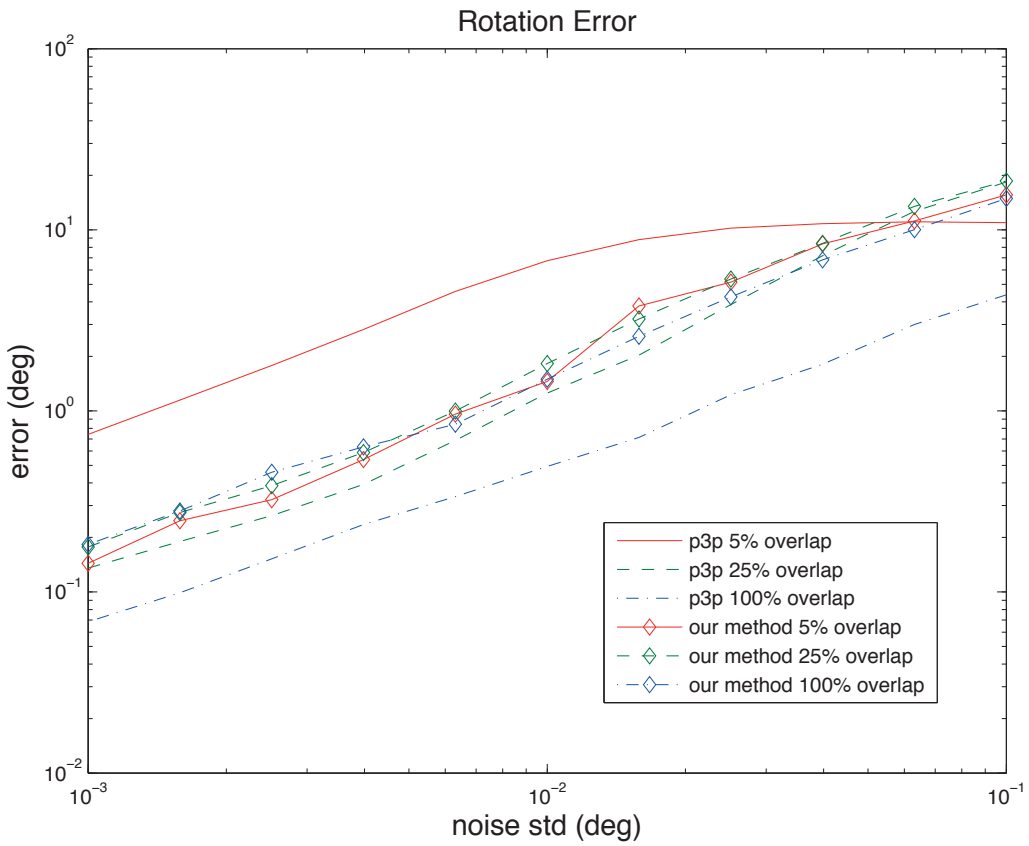

Figure 4.10: Comparison of absolute rotation error after RANSAC without outliers. This is the error for the best sample from RANSAC without non-linear refinement.

the cameras. The cameras have a 60° field-of-view horizontally and vertically.

Comparing Figures 4.10 and 4.11 one can clearly see that the performance of the P3P method decreases with decreased overlap in the cameras while our method has virtually constant performance regardless of overlap. With 100% overlap, P3P outperforms our method. However, with 25% overlap, the two methods perform comparably and with 5% overlap, our minimal solution outperforms P3P for typical noise values. Figure 4.12 shows that our method performs with roughly the same scale error regardless of overlap while the P3P method degrades.

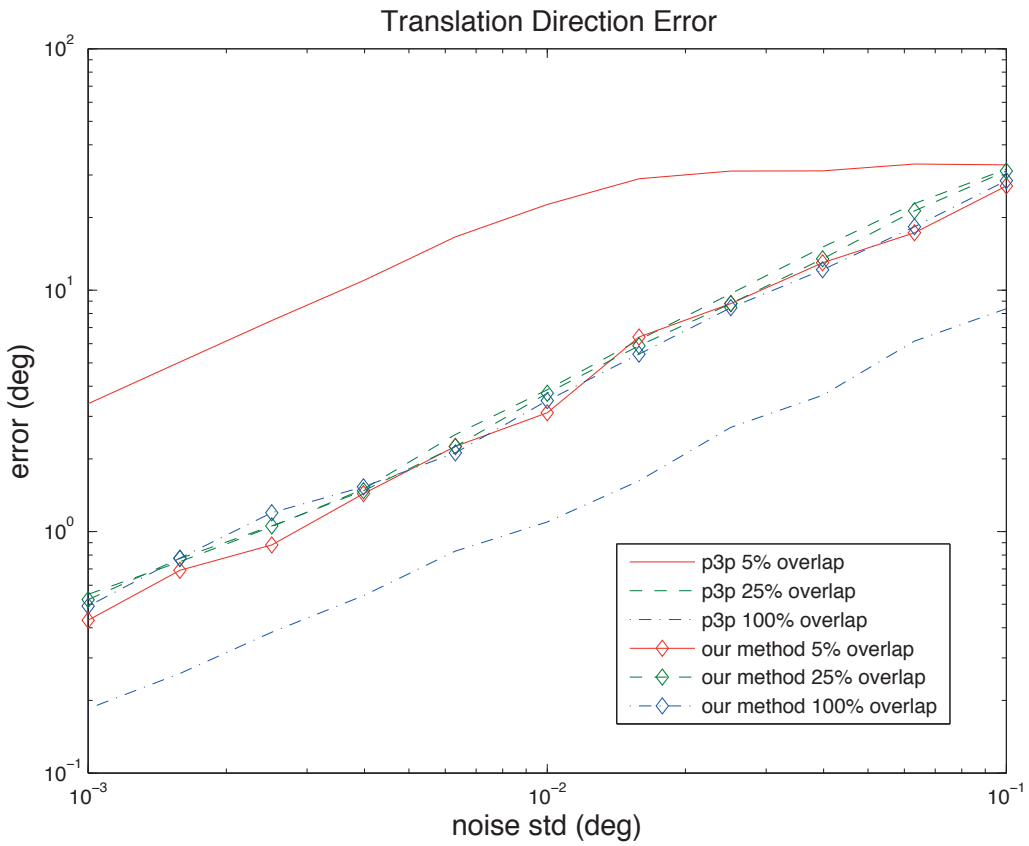

Figure 4.11: Comparison of translation direction error after RANSAC without outliers. This is the error for the best sample from RANSAC without non-linear refinement.

4.8 Experimental Evaluation in Structure from Motion

To demonstrate our minimal solver we have incorporated it into a real-time (12fps processed), stereo camera based structure from motion system. The system uses a stereo pair of 1024x768 resolution cameras with approximately 40^o by 30^o fields of view to collect video input. The system does 2D feature tracking using a graphics processing unit (GPU) implementation of multi-camera scene flow (Devernay et al., 2006). This work is an extension of Kanade-Lucas-Tomasi (KLT) tracking (Lucas and Kanade, 1981) into three dimensions. Features are matched between the two cameras in the stereo head and triangulated. Image motion can then be parameterized as the motion of the 3D feature in front of the camera. This gives accurate correspondences between the four cameras of a stereo pair at

two different times. In addition, we also track features that cannot be matched between the left and right image using a standard image space KLT formulation. The image feature tracking runs at approximately seventy frames per second including stereo feature matching and feature re-detection.

Estimating the scene structure and camera motion is done using a RANSAC framework to find an initial pose estimate, followed by a local bundle adjustment to refine the camera poses and 3D feature estimates. Structure from motion is performed only on key-frames, which are determined based on the average feature motion in the images. This considerably speeds up processing of a video sequence without significant loss in accuracy. The RANSAC framework uses the minimal solver described in this chapter and makes the modifications to RANSAC mentioned in Section 4.4. Local bundle adjustment is performed on the previous seven key-frames and all of the features visible in at least two of those views. Additionally, the two least recent camera poses are held fixed to ensure the continuity of the SfM results.

Figure 4.13 shows a top down view of the left camera path calculated using a video sequence shot in an office environment. Example images of the video sequences are show in Figure 4.14. The office loop is approximately 18 by 10 meters. The camera rounded the loop twice. The path was not exactly the same in both trips around the loop, which accounts for most of the variation of the paths. Note the upper left of Figure 4.13, where the camera path crosses over itself three times just before the clockwise turn. This location is a point of constriction in the environment, which forced the camera to take the same position on each trip around the loop and is shown in the top right image of Figure 4.14.

4.9 RANSAC with Multiple Solvers

Typically, in solving for a stereo camera's motion using RANSAC one solution method is selected before hand and then applied to the available correspondences. Selecting a solver

beforehand, without knowledge of the correspondences available may not produce optimal results. For example, say that four-view features have a very low inlier ratio for some reason in a given sequence. However, three-view features have a very high inlier ratio. Then selecting case$< 1, 0, 3 >$ as the solution method would be much less efficient than the P3P method, case$< 0, 3, 0 >$.

What is needed is a modified RANSAC approach that can find the best solver to use, given the inlier ratios found in a video sequence. These ratios can be calculated with windowed averaging over the last few frames. Then, given these inlier ratios finding which solver to try next in a RANSAC procedure is fairly simple. We simply choose the solver which after it has run will most increase the probability of the solution being correct as shown in Equation 4.8. In that Equation j, k, and l are the number of four-view, three-view and two-view features in a given solver and i_n is the inlier ratio for n-view features. $|case < j, k, l >|$ then is the number of times the solver using j four-view, k three-view and l two-view features has been tried in the RANSAC iterations so far. The function $C(j, k, l)$ returns the number of times the solver for $case < j, k, l >$ has been used so far in the RANSAC iterations.

$$\max_{case<j,k,l>} \left(\left(1 - i_4^j i_3^k i_2^l\right)^{C(j,k,l)} - \left(1 - i_4^j i_3^k i_2^l\right)^{C(j,k,l)+1} \right) \tag{4.8}$$

In this framework we can also take into account the relative cost or calculating each minimal solution. The five-point method (and so case$< 0, 1, 4 >$) is perhaps a factor of twenty slower to compute than the P3P method based on my experiments. If we do not take this higher cost into account then we would not find the correct solution in the fastest possible time. The RANSAC solution method selection equation which takes into account computation cost is shown in Equation 4.9. The next solver selected in the RANSAC is the solver which maximizes this equation. In the equation $cost_{case<j,k,l>}$ is the time taken to compute a solution of $case < j, k, l >$.

$$\max_{case<j,k,l>} \left(\frac{\left(1 - i_4^j i_3^k i_2^l\right)^{C(j,k,l)} - \left(1 - i_4^j i_3^k i_2^l\right)^{C(j,k,l)+1}}{cost_{case<j,k,l>}} \right) \tag{4.9}$$

Of course, Equations 4.8 and 4.9 require knowledge of the inlier ratios for the various types of features. If these inlier ratios are unknown then a sort of Catch-22 appears, a logical paradox arising from a situation in which an individual needs something that can only be acquired by not being in that very situation. In order to select the best solver to get the solution as fast as possible we need to know the inlier ratios. However, until we find the correct solution we have no knowledge of the inlier ratios. For this reason, RANSAC which selects between solution methods can only be applied to situations where the inlier ratios are at least roughly known a-priori. After the first frame, visual odometry on a video sequence meets this requirement, since the inlier ratios from previous frames can be used as an approximation of the current frame's inlier ratios.

4.10 Conclusion

In this chapter, we have introduced a novel minimal solution for the relative pose of a stereo camera using one feature observed in all four views, two two-view correspondences from the left (or right) camera and one two-view correspondence from the other camera. Our approach allows the scaled translation to be estimated between poses while at the same time enables a wide total field-of-view to increase the relative motion estimation accuracy. We have evaluated our solver on synthetic data with noise and outliers. Additionally, we demonstrated our solver's application in a real-time visual odometry system.

This chapter completes the discussion of our work on motion estimation for rigid multi-camera systems. The next chapter will introduce a parallel, real-time VSLAM system. This system takes advantage of the underlying parallelism in the VSLAM problem to enable the exploration and mapping of areas the size of a small office building online and in real time.

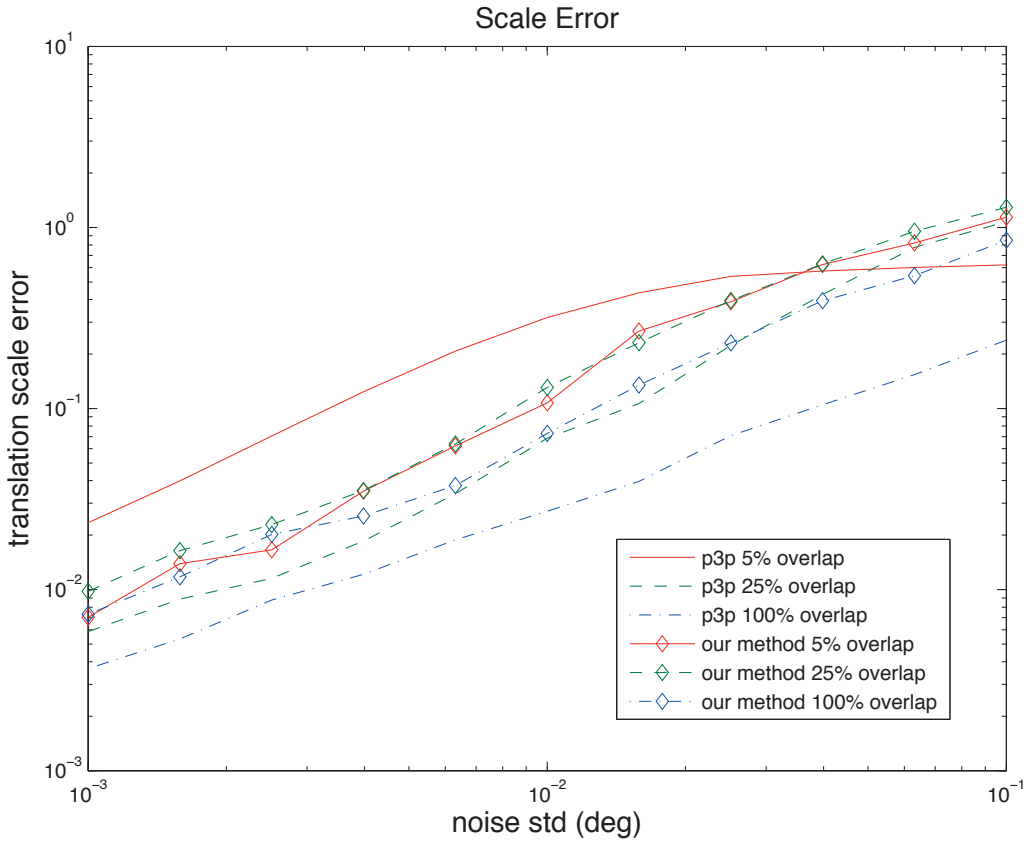

Figure 4.12: Error in the length of translation direction after RANSAC without outliers for out solution method and P3P with varying stereo camera overlap. No refinement is performed on the best RANSAC sample.

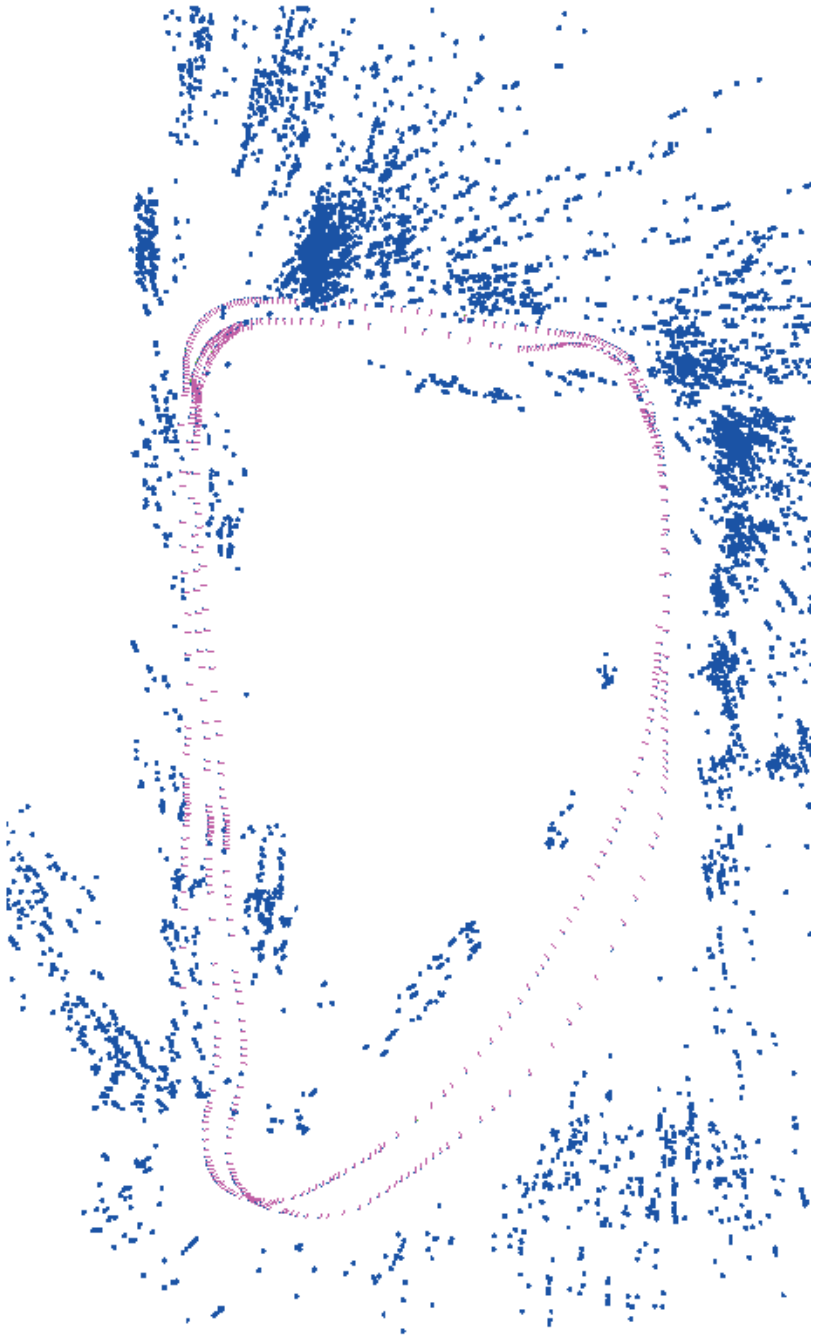

Figure 4.13: View from above of the reconstructed camera path showing the overlapping loops. The camera made just over two roughly 18x10m laps around an office environment. No global bundle adjustment was performed. We have attempted to remove features on the ceiling and floor so the layout of the environment is visible. Left camera axes are drawn as well as a purple line for the baseline.

Figure 4.14: Sample frames from the left camera of the stereo pair for the office sequence. The images are ordered left to right, top to bottom according to their time in the sequence.

CHAPTER 5

Real-Time Globally Euclidean VSLAM

5.1 Introduction

In recent years the visual simultaneous localization and mapping (VSLAM) problem has become a focus of the robotics and vision communities. This effort has been made possible by advances in camera hardware and the computational power available in a personal computer. In this chapter, we introduce a novel, real-time system for six degree of freedom visual simultaneous localization and mapping. Our system fully decouples Visual Odometry and Global map correction. This decoupling enables our system to create Euclidean maps of larger areas than have been mapped before in real-time. Real-time operation at more than 15 frames per second is achieved by leveraging a combination of data parallel algorithms on the GPU, parallel execution of compute intensive operations and producer/consumer thread relationships that effectively use modern multi-core CPU architectures.

Indoor environments, due to their lack of salient features, pose a particular problem for visual navigation. A combination of local tracking and global location recognition enables our system to robustly operate in these environments. The system is demonstrated on two challenging indoor sequences that include sections with very few salient features to track because of large textureless regions. To overcome inherent drift problems from local feature tracking the system detects loops once it re-enters an area it has mapped before using SIFT (Lowe, 2004) features. The loop closing mechanism additionally enables re-initialization into the global model after local tracking failure. We demonstrate the im-

provement in the maps after loop detection and loop completion in comparison to using only visual odometry, which does not detect loops.

5.2 System Description

Our parallel, real-time VSLAM system is composed of three primary modules: Scene Flow (SF), Visual Odometry (VO), and Global SLAM (GS) as shown in Figure 5.1. The Scene Flow module calculates the sparse optical flow and selects key-frames based on the magnitude of the average flow. It then passes the key-frames and the tracks to the Visual Odometry module, which calculates the inter-key-frame motion and passes this motion as well as the 3D features to the Global SLAM module. The Global SLAM module then performs loop detection, and global error correction of the map based on the detected loops. The final result of our method is a globally consistent sparse 3D model of the environment made up of 3D feature points and camera poses for the key-frames.

5.2.1 Scene Flow Module

To determine the local motion of the camera we track features from frame to frame using multi-camera scene flow proposed by Devernay et al. in (Devernay et al., 2006), which is an extension of differential KLT tracking into three dimensions. To meet the real-time goal our system uses an efficient GPU based implementation. In multi-camera scene flow, features are first extracted using the approach of Shi and Tomasi (1994) and then matched from left to right image enforcing both the epipolar constraint and cross-validating feature matches to eliminate outliers. After the features are matched, they are triangulated to establish 3D positions for the inlier features. At this point, features are tracked as small 3D planar surface patches in front of the cameras. The feature motion in 3D is determined through the temporal image flow of the 2D features in the stereo cameras. Using this parametrization the epipolar constraint is enforced without resorting to stereo matching a feature in each

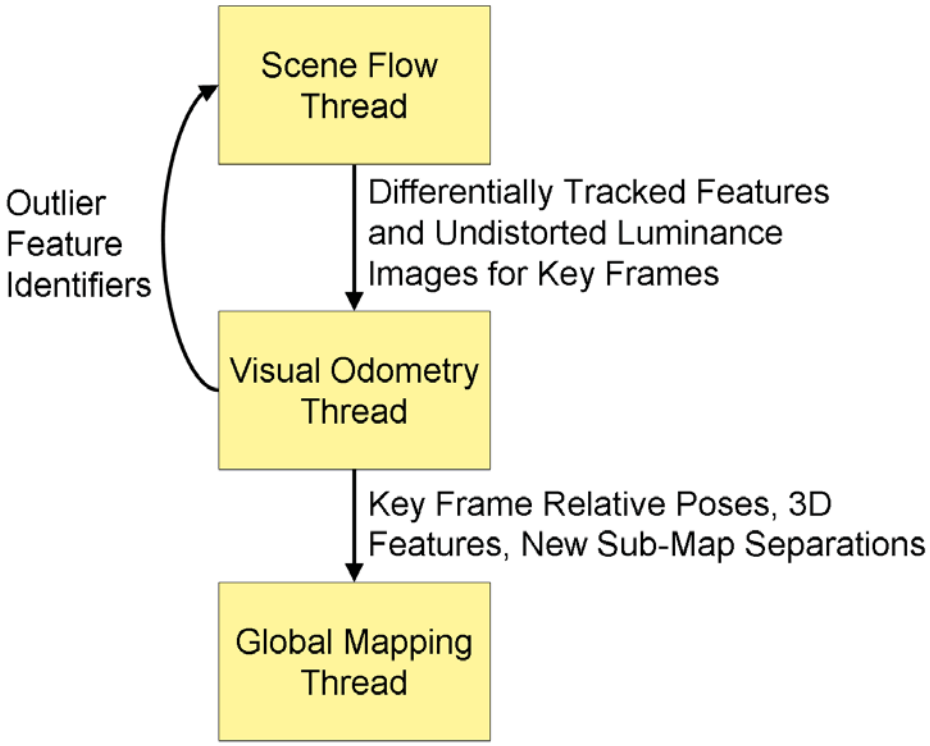

Figure 5.1: The main threads of the system architecture.

stereo image.

Given the varying temporal redundancy in the video, which is mainly due to camera motion, we adaptively select key-frames through a threshold on the minimum average optical flow of features since the last key-frame. To minimize costly feature detection, detection is only performed in the selected key-frames with the additional constraint that if too few features are tracked, another key-frame occurs. Hence, images with small camera motion are not taken as key-frames. The 2D feature tracks and the triangulated 3D points are then passed to the Visual-Odometry module.

5.2.2 Visual Odometry Module

The stereo camera system enables estimation of the 3D points in the Scene Flow module. Therefore, we use an approach along the lines of Nistér (2004) to determine the camera motion. Our method uses the three-point perspective pose method (Haralick et al., 1994)

in a RANSAC (Bolles and Fischler, 1981) framework to determine the camera pose using tracks from the Scene Flow module. While this method is sufficient for tracking the differential camera motion it accumulates small inaccuracies over time, which theoretically lead to an unbounded drift.

To counter the drift our system detects camera path intersections using SIFT features (Lowe, 2004). SIFT features can be matched over large changes in viewpoint, in contrast to differentially tracked KLT-features. To boost performance we use a CUDA based GPU implementation (Wu, 2007). In addition to using the SIFT features for loop detection we also use them in refining the incremental motion estimation. This refinement is performed using a windowed bundle adjustment (Engels et al., 2006) delivering refined camera poses and more accurate 3D points than those delivered by the scene flow described in Section 5.2.1. In our windowed bundle adjustment a window spanning the last n (typically seven) key-frame poses is optimized. We selected seven key-frames in our windowed bundle adjustment because it seemed to be a good point in the trade space between computation speed and mapping accuracy. This trade space is explored in detail in Engles et al. (2006). The oldest two key-frame poses are held fixed while the youngest $n - 2$ key-frame poses are varied along with all of the 3D features, both SIFT and KLT. The bundle adjustment uses a robust cost function so outliers have a limited influence on the result.

Combining the refined camera motion estimate based on KLT feature tracks with the 3D position of the SIFT features we can predict where the SIFT features should project into the current key-frame. We use this prediction to our advantage by limiting the candidate matches to close by SIFT features in the current key-frame (see Figure 5.2). The benefits of this are twofold. We are less prone to problems caused by repetitive structures and given the smaller number of potentially matching features we can reduce the number of SIFT-descriptor comparisons. Furthermore, we empirically found that this prediction allows us to relax Lowe's SIFT matching uniqueness criteria (Lowe, 2004) but still be robust to repetitive structures in the scene.

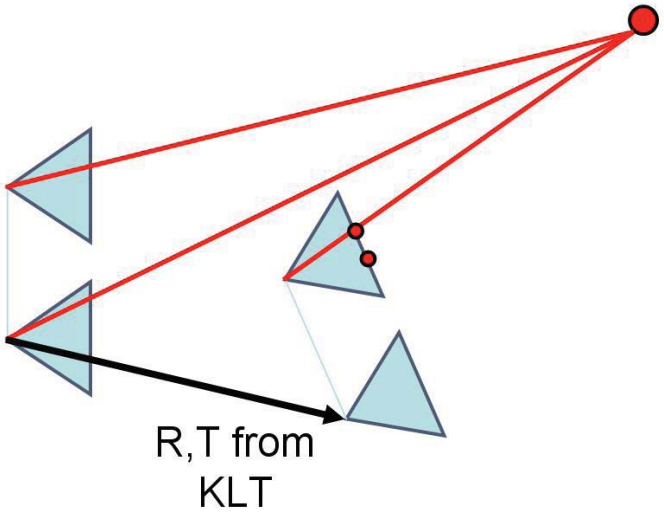

Figure 5.2: Repetitive SIFT features can be disambiguated using geometry calculated from the scene flow features. Additionally, using the geometry reduces the number of potential SIFT feature correspondences tested increasing matching efficiency.

Following predictive SIFT matching we match the remaining unmatched SIFT features from left to right images in the current key-frame using the stereo camera's calibration to constrain the search for matches along the epipolar lines. These matches are then triangulated and un-matched SIFT features are discarded.

At this point, the newest key-frame has been completely incorporated into the local map. It will be considered until it leaves the bundle adjustment window or the visual odometry fails and a new sub-map is started. Please note that as soon as a frame has an initial pose in the visual odometry module, its 3D pose with respect to the global map can be found. This pose will be locally accurate and will be refined through the windowed bundle adjustment. The pose may be changed when loops are detected in the Global SLAM module but this should not affect tasks such as obstacle avoidance. After exiting the bundle adjustment window, key-frames are processed by the Global SLAM module.

5.2.3 Global SLAM Module

The Global SLAM module ensures global consistency in our VSLAM system. It incorporates the information of all currently available key-frame poses, feature measurements, and initial 3D feature estimates from the Visual Odometry module. The final result is a set of globally consistent, Euclidean sub-maps each of which has its own global coordinate frame. The sub-maps are disjoint, meaning they cover separate areas in the environment or cannot be linked by common 3D features due to limitations of wide-baseline feature matching.

The key element to improve the incremental motion estimation provided by the Visual-Odometry module is the detection of loop completions. Loop completions provide additional constraints to the local constraints found in the VO module. Our system uses the vocabulary tree (Nistér and Stewénius, 2006) based approach to detect loops. Please note, any alternative approach like the Fab-Map approach of Cummins and Newman (2008) could be used instead. In our approach, SIFT feature descriptors are quantized into visual words using a K-d tree over a descriptor space, which is pre-computed. The visual words seen in an image are then organized so that one can find out quickly, which images a visual word is seen in. Finding similar images to a query image is then as simple as computing a vote to determine in what other images a query image's visual words are found. In the vote higher weight is given to the more discriminative visual words that are found less frequently.

The Global SLAM module can operate in one of two modes. When exploring new areas the system operates in *loop seeking mode* while in previously mapped regions the system operates in *known location mode*.

5.2.3.1 Loop Seeking Mode

Loop seeking mode performs loop detection for each new key-frame and after a successful loop identification a global refinement is computed through bundle adjustment. Loop de-

tection begins by using the vocabulary tree to find a list of the most similar images to the current key-frame sorted by similarity. Images from recent key-frames are removed from the list so that loops are only found to older sections of the map. In our system recent key-frames are selected by the number of key-frames between the current key-frame and other key-frames and is a selectable parameter. A more principled approach might use visibility constraints on the previous key-frames in the sequence to determine which ones could see the current key-frame's features and therefore should be considered "recent". Images in the list are tested in order of similarity until a matching image is found or the similarity score of the next best match is too low.

Rather than simply match SIFT features from the query image to those visible in the next most similar image we use the putative matching image to find a region of local 3D scene structure and match the query image to this structure. This can be seen as a form of query expansion based on 3D geometry. The expansion is done by finding the images near the next most similar image and including all of the 3D features visible in all of these images in the SIFT matching and geometric verification. The SIFT matching is then performed from the image to the 3D structure. SIFT descriptor matching is performed from the descriptors of the features in the current key-frame to the 3D features' descriptors. We only try to match SIFT descriptors with the same associated visual word, which reduces the number of descriptor dot products performed. A RANSAC process using the three-point perspective pose method is then used to find the pose of the current camera and the pose is non-linearly optimized afterwards.

If the above method finds a solution supported by enough inlier matches, it is considered a loop. The features associated with the inlier measurements to the RANSAC are linked so that they are treated as a single feature in bundle adjustment. Using 3D feature to 2D projection matching with geometric verification makes false positive loop detections much less likely than using an image-to-image, appearance only, matching approach. This is because our approach combines both visual and geometric similarity to detect loops. Still

truly repetitive 3D structures which also look the same can cause incorrect loops to be detected. Dealing with repetitive structures remains an open research problem.

If no loop has been detected, then the next key-frame is tested for a potential loop closing. If a loop was detected, the system performs a global correction to the current sub-map incorporating the newly detected loop. Since the newly detected loop features have high reprojection errors in the current key-frame, they would be deemed invalid by our bundle adjustment, which uses a robust cost function. Hence, they would not influence the error mitigation process. To overcome this effect we re-distribute the error before bundle adjustment. This initializes the bundle adjustment much closer to the global minimum of its cost function, increasing its convergence rate and decreasing the chance of converging to a local minimum.

We re-distribute the accumulated error by starting with the difference in the current key-frame pose and the current key-frame's pose calculated w.r.t. the old features. This gives us the amount of drift that the system has accumulated since it left the last known location in the sub-map. This last known location is either the first frame in the sequence if no loops have been found so far or the last place the system was operating in known location mode. The system is operating in known location mode when it has reacquired features it has mapped before and is tracking with respect to that known map. The system linearly distributes the error correction for the cameras back to the point it was operating in known location mode. Spherical linear interpolation (Shoemake, 1985) of the rotation error quaternion is used to interpolate the rotation error. Feature points are similarly corrected by moving them along with the camera that first views them. A global bundle adjustment of the map is then performed. After bundle adjustment, outlier measurements are removed as well as features visible in fewer than two key-frames. These features give little information about the scene structure and are more likely to be incorrect since they do not match the camera's motion. After successfully detecting the loop and correcting the accumulated error the Global SLAM module enters known location mode.

5.2.3.2 Known Location Mode

After successfully identifying a loop, this mode continuously verifies that the robot is still moving in the previously mapped environment. Verification is done by linking the current 3D SIFT features to previously seen 3D SIFT features in the environment surrounding the current location. These matches are added to a windowed bundle adjustment in the GS module, which keeps the camera path consistent with the older previously computed parts of the map.

In known location mode SIFT feature matching between the current key-frame and the old 3D SIFT features is done using the predictive approach described in the visual odometry module (see Section 5.2.2 for a discussion of visual odometry). Older features can be linked to the features visible in the current frame by projecting all of the 3D SIFT features seen in the previous key-frame and it's neighboring images (two key-frames are neighbors if they see the same 3D feature) and comparing descriptors. If no matching older SIFT features are found then the robot has left the previously observed parts of the environment and the system reenters the "Loop Seeking" mode.

The windowed bundle adjustment in GS is much the same as the one performed in the Visual Odometry module. The only difference in this case is that the older key-frames are also included in the bundle but fixed. This ensures the new camera poses stay consistent with the existing map. Fixing the older cameras is also justified since they have already been globally bundle adjusted and are probably more accurate than the more recent key-frames. After the windowed bundle adjustment processing begins on the next key-frame.

5.3 Implementation Details

A key to the performance our system is that each of the three modules Scene Flow, Visual Odometry, and Global SLAM operates independently and in parallel. To ensure that all captured information is used, only the Scene Flow module has to operate at frame-rate. The

timing constraints on the visual odometry are dynamic and only depend on the frequency of key-frames. This module can lag behind by a few frames. The Global SLAM module is less time constrained since its corrections can be incorporated into the local tracking when they are available. The system's modules operate in separate threads that each adhere to the individual module timing requirements.

5.3.1 Scene Flow Module

The Scene Flow module begins by taking raw, Bayer pattern images off of the stereo cameras. These images must be converted to luminance images and radially undistorted before the sparse scene flow can be measured. We use color cameras so that the video we record can later be used for dense stereo estimation and 3D modeling. While tracking could be performed on radially distorted images, we remove the radial distortion from the images so that later SIFT feature extraction in the Visual Odometry module can be done on undistorted images. Using undistorted images helps in SIFT matching when using cameras with a large amount of radial distortion.

De-mosaicing, radial undistortion, and sparse scene flow are all calculated on the graphics processing unit (GPU) using CUDA. To increase performance we minimize data transfer between CPU to GPU by downloading the raw image to GPU for each frame, performing all computations in GPU memory, and then only uploading undistorted images to the CPU for the key-frames as well as the tracked feature positions.

After each key-frame the feature tracks (2D position and feature identifier) and the undistorted images are passed to the Visual Odometry module. While the Visual Odometry module processes the key-frame the Scene Flow thread can track ahead of it, buffering new key frames until the Visual Odometry module is able to process them. Hence, the speed of Visual Odometry does constrain the Scene Flow module's real-time performance. This is just one example of how parallelism adds robustness to our system.

5.3.2 Visual Odometry Module

In this module we perform the incremental motion estimation from the KLT-features tracks and the detection of SIFT features in parallel. For efficiency we use one thread for each of the two stereo images. After the SIFT detection we release the image buffers to save memory.

As described in Section 5.2.2 the Visual Odometry module's outputs are the relative camera motion and the new 3D points. These outputs are stored in a queue and are removed from Visual Odometry's local storage. Using a queue decouples processing in the VO and GS module threads. Whenever tracking fails all the VO module's internal data (key-frame poses and 3D features) is queued for processing by the Global SLAM module.

5.4 Experimental Results

In order to demonstrate the speed, accuracy, and long-term stability of our VSLAM system we present results from two video sequences of two indoor environments with different characteristics. The first sequence was taken in an office environment, which has a large, open floor plan. I will refer to this as the "office" sequence. The second sequence was shot in a building with long, but relatively narrow (1.7m) hallways. It will be called the "hallway" sequence. The closed floor plan does not allow features to be tracked for long periods of time since they quickly leave the stereo camera's field-of-view, yet the system successfully maps the halls accurately with an error of less than 30cm over the 51.2m length of the longest hall shown in Figure 5.10. This is an error of less than 0.6%.

Our setup uses a calibrated stereo camera pair consisting of two Point Grey Grasshopper cameras with 1224×1024 pixel resolution color CCD sensors delivering video at fifteen frames (stereo pairs) per second. The system's 7cm baseline is comparable to the median human inter-pupil distance. The cameras are mounted on a rolling platform with the computer. Using a rolling platform the planarity of the camera path can be used to evaluate the

quality of the reconstruction results. However, the full six degrees of freedom are estimated for the camera's motion. While performing real-time VSLAM the system also records the imagery to disk for debugging or archival purposes.

The office sequence includes transparent glass walls and other reflective surfaces that make tracking more challenging (please see Figure 5.3 for example frames). It also has a hallway with relatively low texture, which our system successfully maps, showing it is robust to areas without a large amount of structure. In one section of the video a person moves in front of the camera, partially occluding some of the tracked features (see Figure 5.3). Even in this case, the system is able to reject the moving person's feature tracks as outliers and continue tracking correctly.

Figure 5.4 shows the difference between operating only using visual odometry and performing the full VSLAM with loop detection and global map correction. In the top pane of Figure 5.4, the map is shown using only visual odometry where the relative motion from frame to frame is accumulated to form the camera path. In visual odometry no loop detection of global map correction is performed hence, the system drifts over time. In this scene, VO accumulated drift of approximately 3m over an approximately 150 meter path. In the bottom pane, the results of our Global SLAM module are shown. Clearly, the long-term drift of visual odometry is eliminated by loop detection and the succeeding error mitigation through bundle adjustment.

Additionally, in Figure 5.5 we show the vertical drift using only visual odometry of approximately 2 meters over a traveled distance of 70 meters. Figure 5.5 shows two side views of the map without (top) and with (bottom) loop detection and global map correction. In the top pane, the accumulated vertical error of 2.0 meters is clearly visible while in the bottom pane it is eliminated. Note the regular pattern on the ground in the bottom pane that reflects the repetitive pattern in the carpet there. We have overlaid the results of our system with loop detection and correction over an architectural layout of the office as illustrated in Figure 5.6. This figure demonstrates the accuracy of our system. The results

shown here were processed from 8304 stereo frames of video at fifteen frames per second including multiple loop detections and global bundle adjustments, one for each time a loop was detected.

We use the hallway sequence to demonstrate our system working in a less open environment where feature tracks are typically shorter and fewer in number. As shown in the panes of Figure 5.8 some of the hallways have large amounts of texture while others are largely textureless. The system robustly performs camera tracking in either case, at times tracking with fewer than ten features.

Figure 5.10 shows an architectural drawing of the building's hallways with dimensions. From the overlay of our model on top of the floor plan, it is clear that our system creates accurate maps. Taking the difference between the measured center-to-center distance of the longest mapped hallways on the figure and the distances between comparable camera centers in our reconstructed map we find an error of 30cm in the horizontal direction and 40cm in the vertical direction in the figure. These errors equate to 0.6% error in the figure's horizontal direction and 1.4% in the vertical.

The three modules' timing results on this sequence are shown in Figure 5.7. Note the four spikes in the Global SLAM module's processing time. These are times when loops were detected and the map was bundle adjusted. In the Scene Flow graph the spikes in processing time occur when new features are detected. In a typical frame 200-500 scene flow features are tracked. In the global SLAM module there are typically 60-120 scene flow features and 100-200 SIFT features, which are inliers to the motion model. Reprojection errors in the Global SLAM module's results average approximately 0.6 throughout the sequence.

Another example of loop completion is illustrated in Figure 5.9 showing the map before loop detection and correction in the top pane and the bottom pane gives the result after loop detection. In these panes, the camera returned to the known area from the right. Note how the accumulated error in the camera poses and features is eliminated by the loop detection

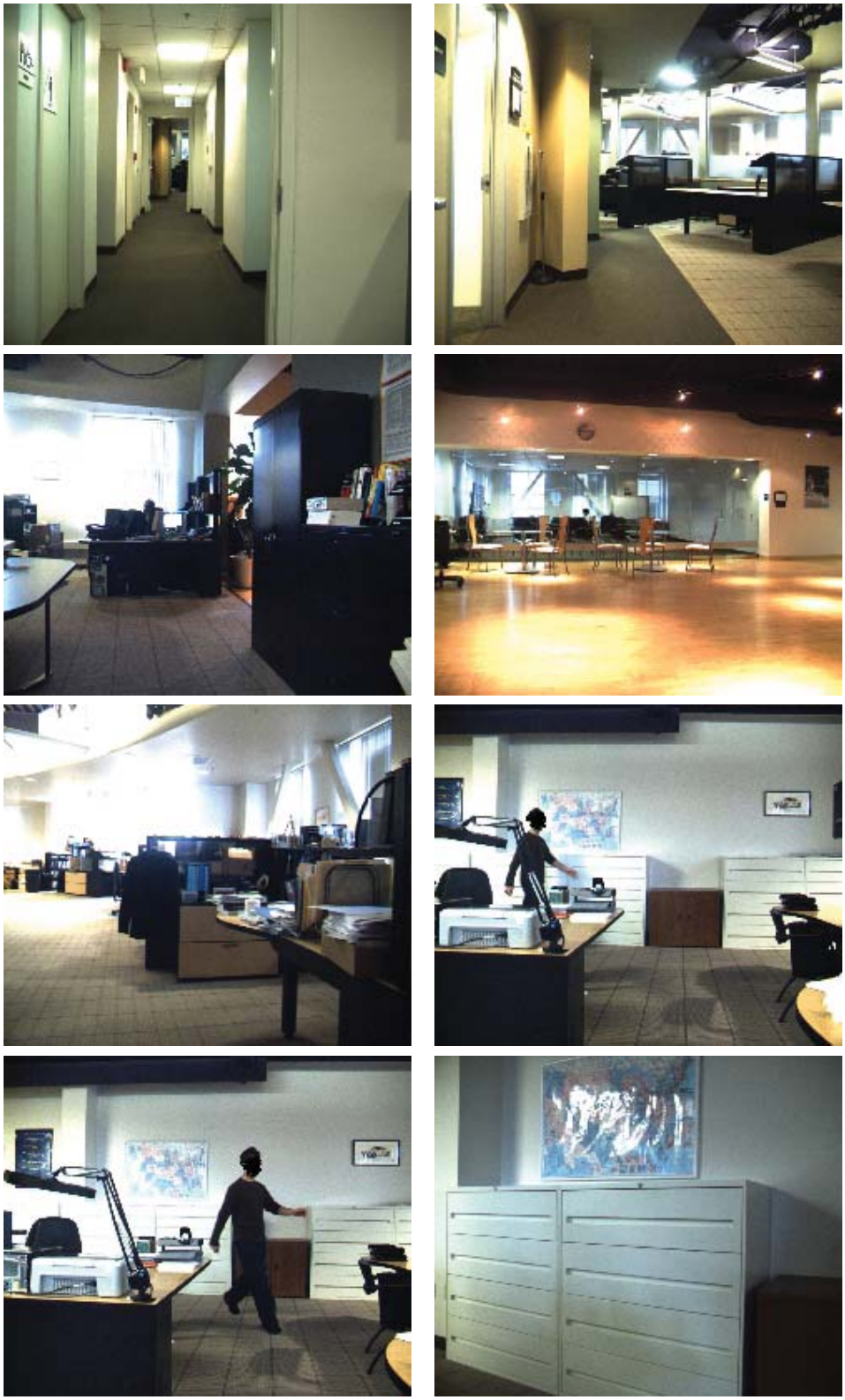

Figure 5.3: Sample frames from the left camera of the stereo pair for the office sequence. Note the reflective glass walls, textureless regions, and moving person in the images.

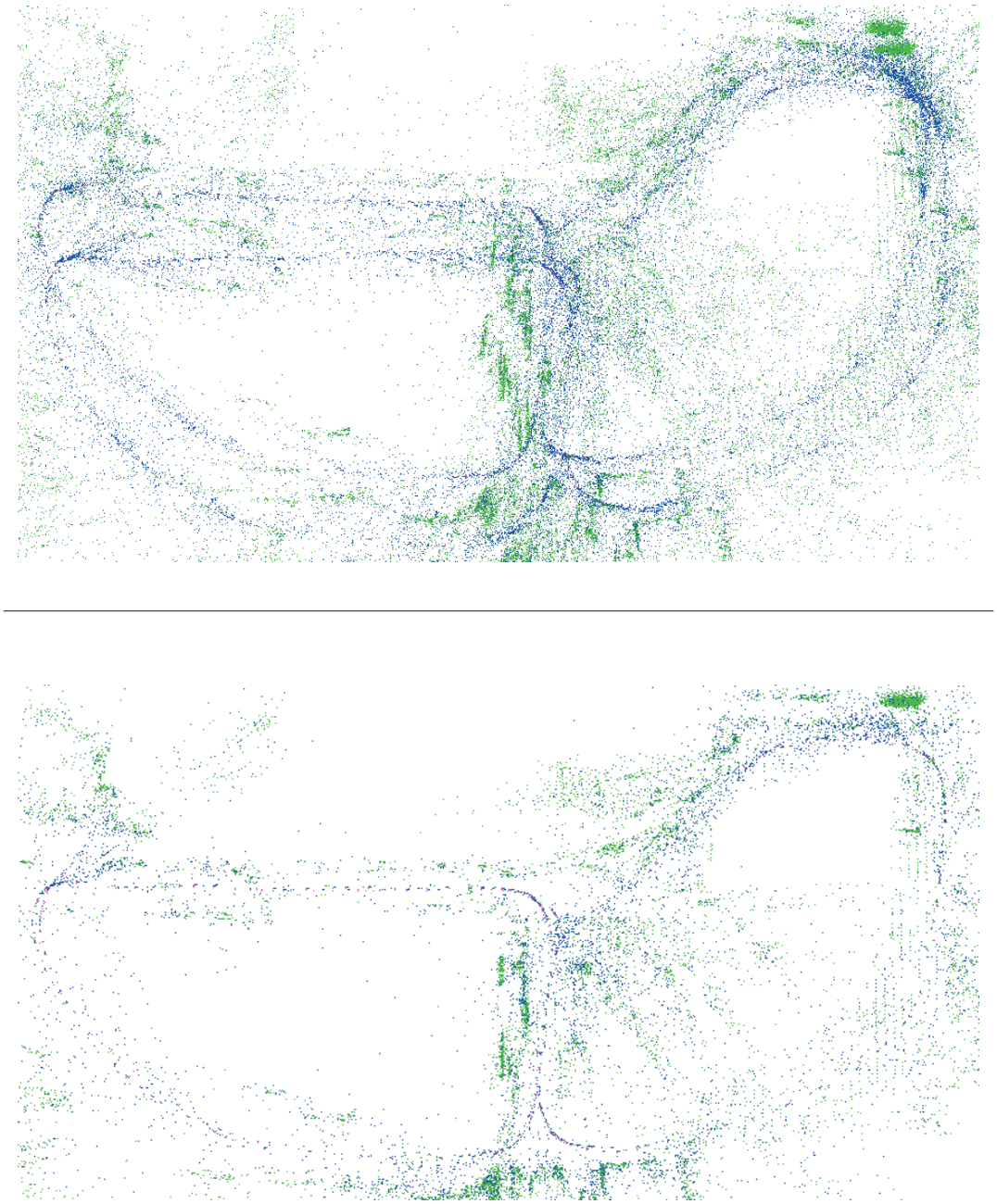

Figure 5.4: Results of office sequence top view. Top: Camera path calculated using visual odometry only. Bottom: Camera path calculated loop detection and correction global slam module. Green points are differentially tracked scene-flow feature and blue points are SIFT features.

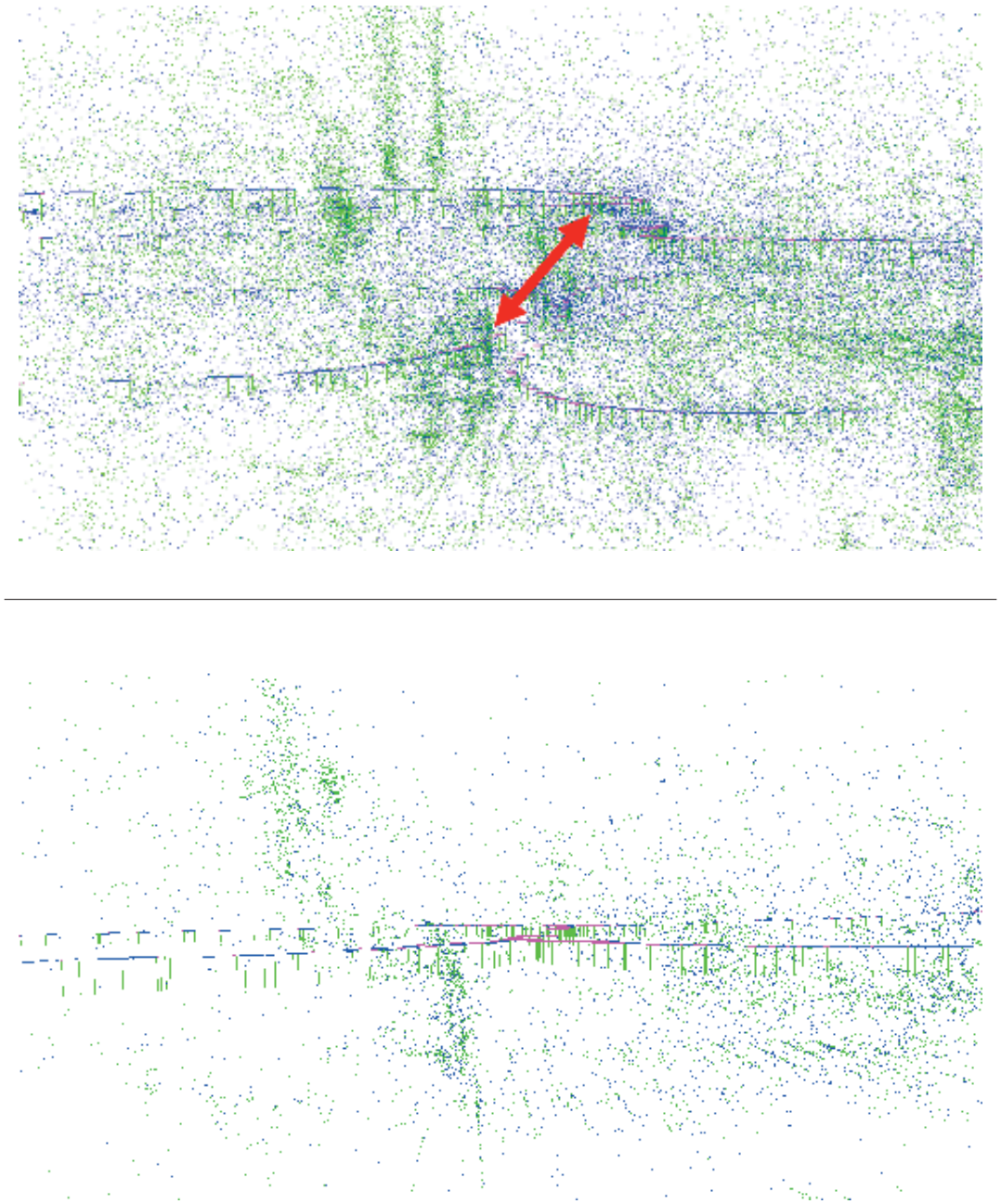

Figure 5.5: Results of office sequence side view. Top: Camera path calculated using visual odometry only. Bottom: Camera path calculated loop detection and correction global slam module. Note that the path processed through the global slam module is planar where visual odometry has large vertical error accumulation as shown by the red arrow. Green points are differentially tracked scene-flow feature and blue points are SIFT features.

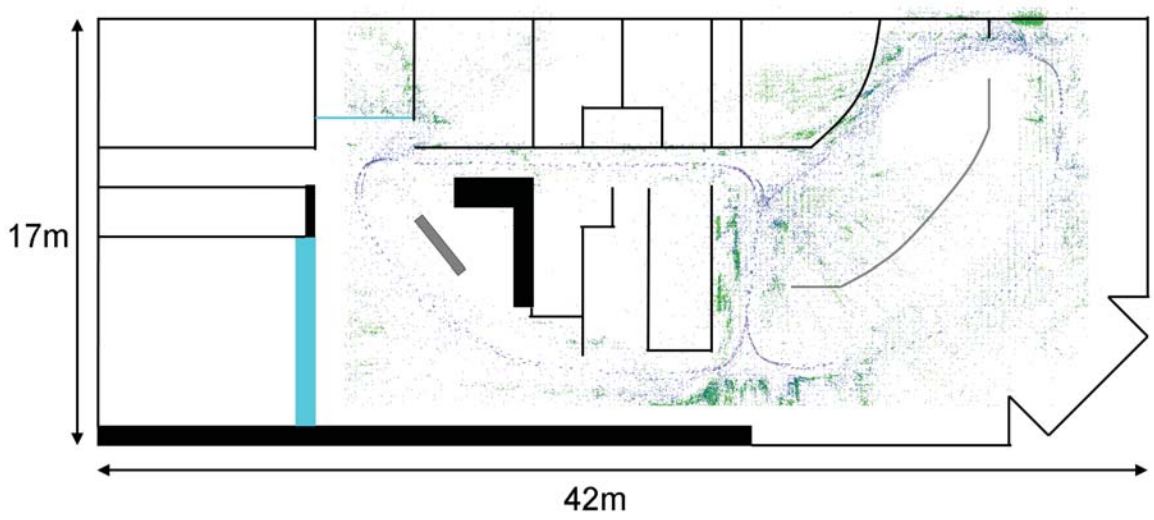

Figure 5.6: Results of office sequence. Camera path from global SLAM module using bundle adjustment for loop correction. The camera made two complete passes around both loops. Blue walls are glass. Grey walls are half ceiling height partitions. Green points are differentially tracked scene-flow feature and blue points are SIFT features.

and global map correction. Finally, a side view of the hallway map is shown in Figure 5.11. The side view demonstrates the planarity of the map, which matches the flat floor of the halls.

5.5 The Challenge of Loop Correction

Creating large-scale Euclidean maps in real-time remains an unsolved challenge in VSLAM. Real-time is a somewhat ambiguous term in this case, so for the purposes here real-time will be considered ”fast enough to support online path planning.” In terms of the multi-threaded approach to VSLAM presented in this chapter, real-time operation means that the system can globally correct the map in the time between subsequent loop completions. This means that real-time operation depends both on the size of the environment and on the complexity of the robot’s path through the environment.

Before delving into a few promising approaches for real-time bundle adjustment, a brief introduction why bundle adjustment scales poorly as the problem size increases is in order.

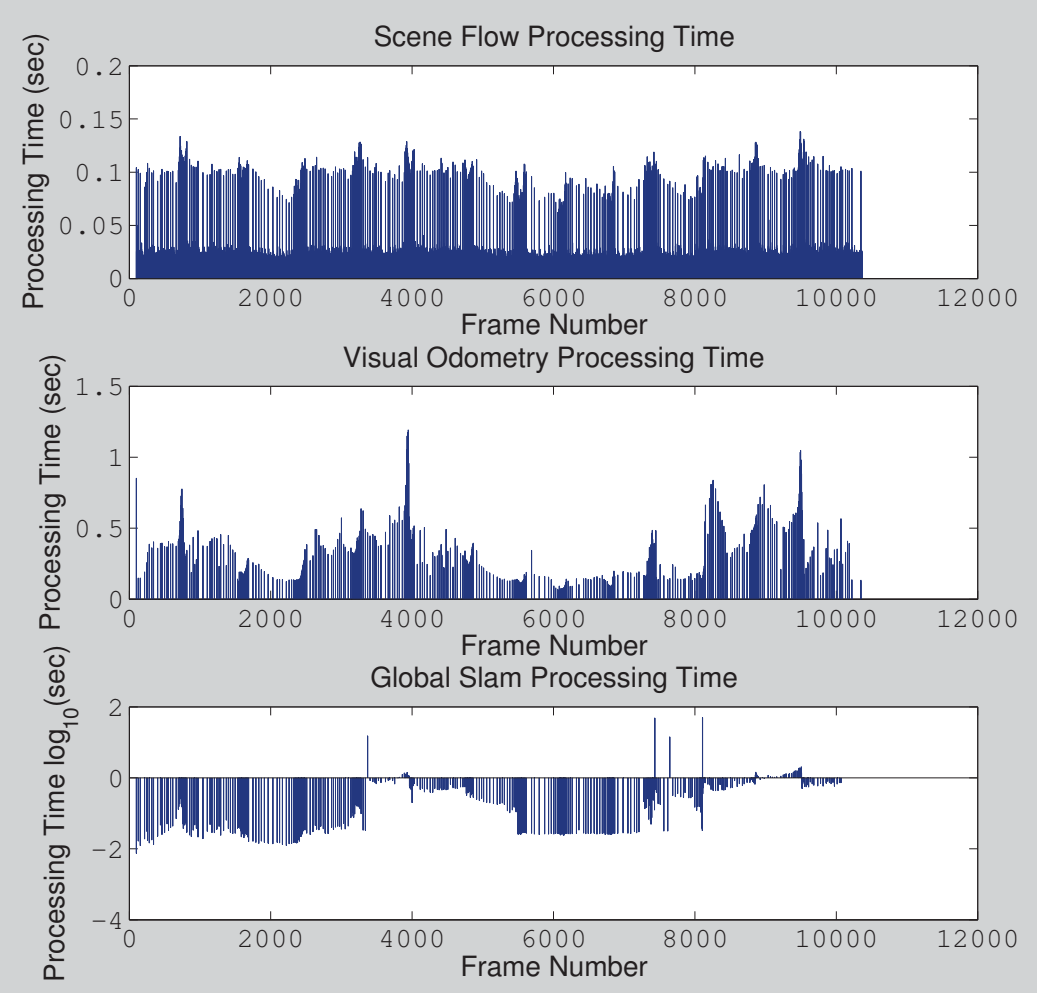

Figure 5.7: Timing results on the hallway sequence. Note the processing time spikes in the scene flow module when new features are extracted. The four large spikes in Global SLAM processing time are caused by bundle adjustments after loop completions. The average scene flow processing rate is 37.1 frames per second (fps), visual odometry is 61.0 fps and global SLAM is 33.5 fps. The VO and Global SLAM processing rates are calculated as total processing time/video frames (not key-frames).

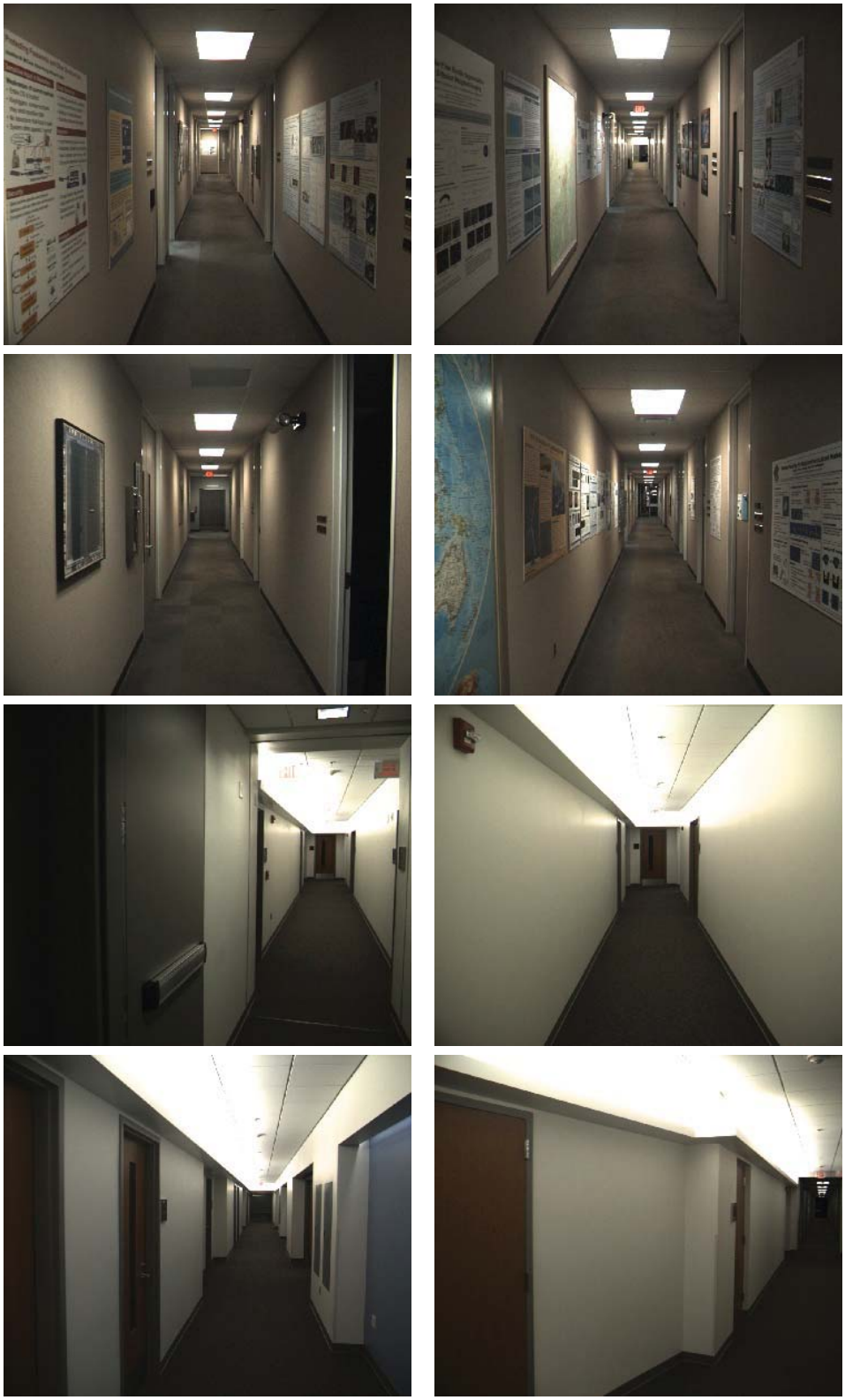

Figure 5.8: Sample frames from the left camera of the stereo pair for the hallway sequence. Note the lack of texture in some images and the forward motion, which makes visual odometry more challenging than if the camera was pointing to the side.

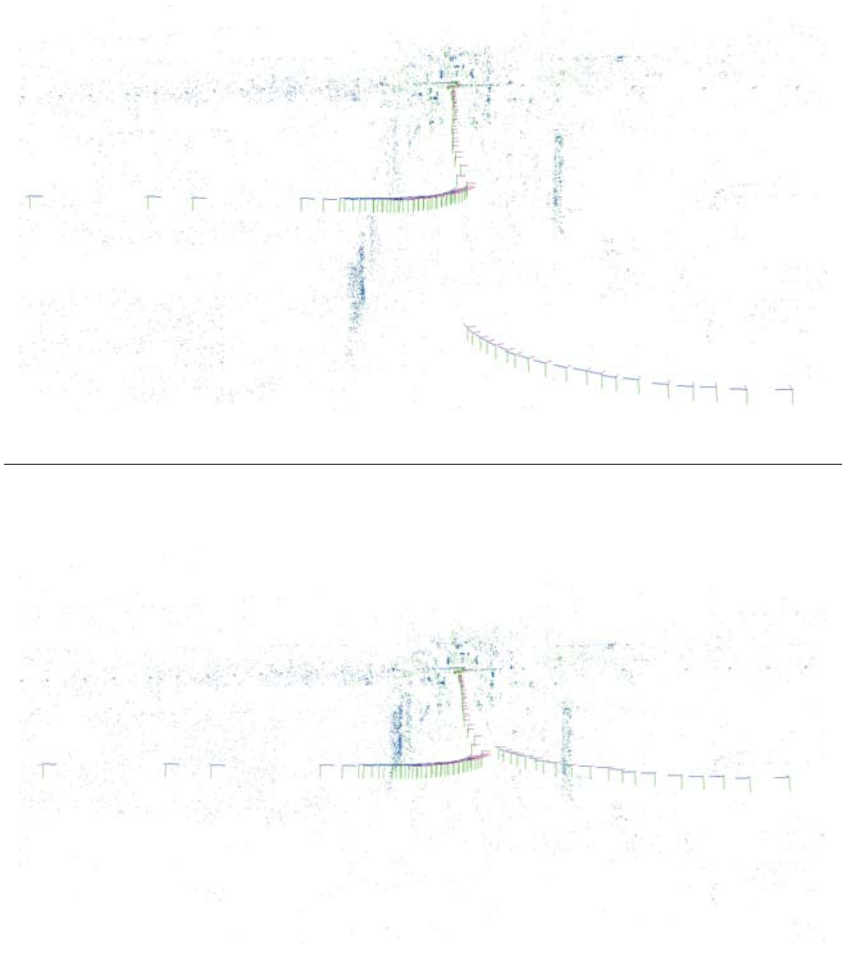

Figure 5.9: Results of hallway sequence. Top: Camera path intersection before loop detection and global map correction. Bottom: Corrected camera path. Note that the paths are now in the same plane. Green points are differentially tracked scene-flow feature and blue points are SIFT features.

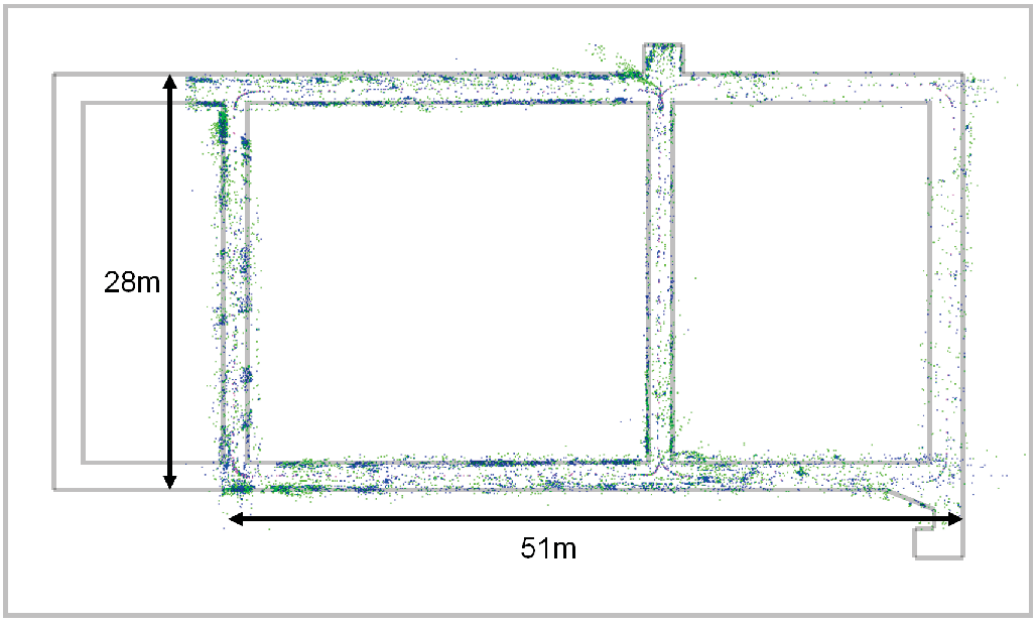

Figure 5.10: Results of hallway sequence. The camera made three rounds of the left loop and one of the right. Green points are differentially tracked scene-flow feature and blue points are SIFT features.

Figure 5.11: The final hallway sequence model viewed from the side. Note that the model is planar matching the planar structure of the building. Green points are differentially tracked scene-flow feature and blue points are SIFT features.

Bundle adjustment uses non-linear minimization to optimize Equation 5.1. This equations assumes that the cameras' calibrations are known. In that equation N cameras and M points are involved in the minimization. R_i and C_i are the rotation and center of camera i. X_j is the 3D location of feature j and x_{ij} is the measured feature j in image i. Π is the projection function and $d()$ is a distance function, typically the reprojection error in pixels.

$$\min_{R_i,C_i,X_j}(\sum_{i=1}^{N}\sum_{j=1}^{M} d(x_{ij},\Pi(R_i,C_i,X_j))^2) \tag{5.1}$$

Equation 5.1 can be expressed as a minimization of the difference of two vectors x, the measured image locations of the features, and $\hat{x}$ the estimated image locations based on the camera pose, 3D feature parameters and camera calibration.

$$x = (x_{11}^T,...,x_{1N}^T,x_{21}^T,...,x_{M1}^T,...,x_{MN}^T)^T \tag{5.2}$$

$$\hat{x} = (\hat{x}_{11}^T,...,\hat{x}_{1N}^T,\hat{x}_{21}^T,...,\hat{x}_{M1}^T,...,\hat{x}_{MN}^T)^T \tag{5.3}$$

Assuming some prior noise distribution on the feature measurements Σ_x bundle adjustment minimizes the squared Mahalanobis distance $(x-\hat{x})\Sigma_x^{-1}(x-\hat{x})$ over the parameters of $\hat{x}$, R_i, C_i and X_j. The Levenberg-Marquadt algorithm is used to minimize the augmented normal equations below.

$$(J^T\Sigma_x^{-1}J+\mu I)\delta = -J^T\Sigma_x^{-1}\epsilon \tag{5.4}$$

In this equation J is the Jaccobian of the projection function Π evaluated at the current state estimate and δ is the change in the state parameters that reduces the residual error $\epsilon = x-\hat{x}$.

$$J_{ij} = (\frac{\partial \Pi(R_i, C_i, X_j)}{\partial R_i} \frac{\partial \Pi(R_i, C_i, X_j)}{\partial C_i} \frac{\partial \Pi(R_i, C_i, X_j)}{\partial X_j}) \mid_{R_i=\hat{R}_i, C_i=\hat{C}_i, X_j=\hat{X}_j} \tag{5.5}$$

In this equation $\hat{R}_i$, $\hat{C}_i$ and $\hat{X}_j$ are the estimates of those parameters for the current iteration of the minimization.

One approach to solve Equation 5.1 is to split the problem into two groups, cameras and 3D features, solve for the camera update and then back substitute to recover the feature updates. This can be done using the Schur complement as shown below.

$$\left(J^T \Sigma_x^{-1} J + \mu I\right) \delta = -J^T \Sigma_x^{-1} \epsilon \tag{5.6}$$

$$\begin{pmatrix} A & B \\ B^T & D \end{pmatrix} = \left(J^T \Sigma_x^{-1} J + \mu I\right) \tag{5.7}$$

$$\begin{pmatrix} \delta_c \\ \delta_p \end{pmatrix} = \delta \tag{5.8}$$

$$\begin{pmatrix} z_c \\ z_p \end{pmatrix} = J^T \Sigma_x^{-1} \epsilon \tag{5.9}$$

$$\begin{pmatrix} A & B \\ B^T & D \end{pmatrix} \begin{pmatrix} \delta_c \\ \delta_p \end{pmatrix} = - \begin{pmatrix} z_c \\ z_p \end{pmatrix} \tag{5.10}$$

In the above equations A is the portion of the augmented Hessian approximation containing the camera parameters, D contains the 3D features and B stores the constraints between features and cameras. δ, the update, has been split into camera, δ_c and a 3D point feature δ_p portions as has the measurement residual mapped into parameter space $J^T \Sigma_x^{-1} \epsilon$.

$$(A - BC^{-1}B^T)\delta_c = -z_c - BC^{-1}Z_p \tag{5.11}$$

$$C\delta_p = -z_p - B^T\delta_c \tag{5.12}$$

The, now decomposed, augmented normal equation is then solved using Equation 5.11 and substituting δ_c into Equation 5.12 to find the point update, δ_p. C^{-1} is easy to compute because it is block diagonal and so this efficiently reduces the least squares problem size.

However, the camera Hessian matrix $(A - BC^{-1}B^T)$ can be fairly dense, making the solution still slow to compute. The more loops there are in the camera trajectory the more fill-in there will be in this matrix. This is because any two cameras that see a 3D feature in common will have a non-zero off diagonal term in the matrix. Block reordering approaches exist to try to make the matrix as diagonal as possible. Ultimately then it is the remaining off-diagonal fill-in that limits the speed of each bundle adjustment update iteration.

Conjugate Gradient approaches have shown some promise in large scale bundle adjustment (Byrod and Astrom, 2009; Agarwal et al., 2010).

Even with more efficient solvers, for large bundle adjustment problems the number of iterations required for convergence grows super-linearly in the number of cameras and features. A collection of images in a bundle adjustment can be visualized as a graph with cameras as nodes and edges connecting cameras with features in common. In the first few iterations local errors in the camera poses and 3D features are reduced fairly quickly. These local error corrections happen quickly because changes to cameras and 3D features need to only move a short distance through the graph to neighboring cameras. At this point what is left is a fairly smooth error function. Corrections to camera poses and features that reduce this error take more iterations to propagate through the graph because they have to pass through a longer path from the node to it's neighbor's neighbors etc. This slow propagation of camera pose and 3D point corrections is shown in Figure 5.12.

To this point the community has addressed this error propagation problem by reducing the size of the bundle. This means reducing the size of the Hessian matrix $J^T\Sigma_x^{-1}J$, which

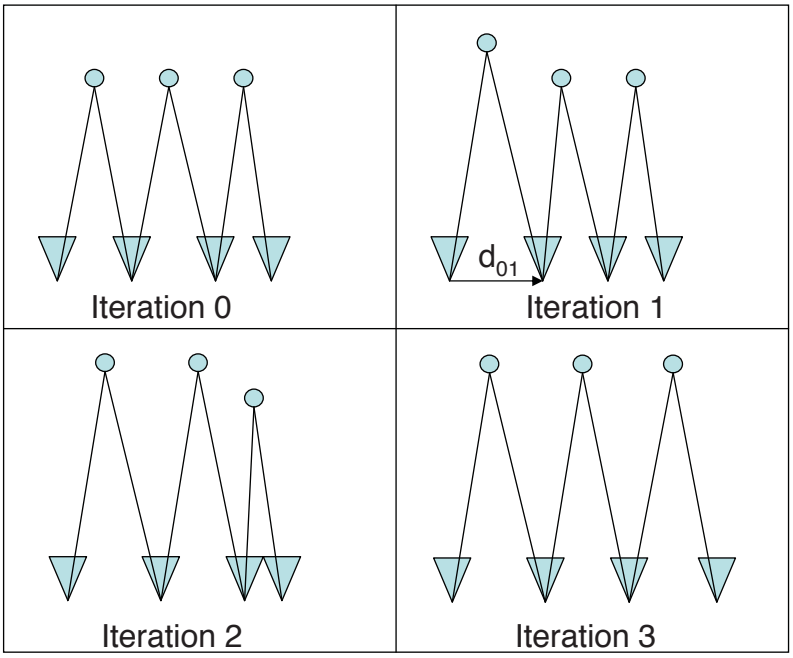

Figure 5.12: The propagation of error through a bundle adjustment problem. At iteration zero the distance d_{01} between the first two cameras is too small (as given by some external measurement system). After iteration 0 the first feature on the left has moved away from the cameras to match the now father apart cameras. This forces the distance between the second two cameras to extend as well as the distance to the second feature and so on. After iteration three the system has reached the minimum of the cost function.

is tied to the number of cameras and the number of features in the map and also limiting the number of measurements. The number of measurements must be reduced because they are involved in calculating the Jaccobians and in the error term ϵ in $J^T \Sigma_x^{-1} \epsilon$.

One approach to reduce the number of cameras and measurements is to subdivide the map into multiple sections, solve them separately and then re-combine them in the final result. This sort of approach is exemplified by the out-of-core bundle adjustment approach of Ni et al. ((2007)) described previously in Section 2.5.4. Out-of-core bundle adjustment has shown some impressive results. However, because it splits the map into multiple section, solves the sections separately, and then re-combines them only optimizing the inter-submap variables, it may not reach the same global minimum that a full bundle adjustment would.

FrameSLAM (Konolige and Agrawal, 2008) presents another possible reduction tech-

nique based on permanently marginalizing out the features from the minimization and possibly some of the cameras. This technique was introduced along with other subdivision-based loop correction techniques in Section 2.5.4 as well. Permanently marginalizing out the features also has its drawbacks. Once marginalized out the feature measurements cannot be re-linearized. This will likely lead to higher error than a full bundle adjustment that re-linearizes the measurements.

What is needed is an approach that makes the large-scale errors propagate through the bundle adjustment graph faster. A scale space approach applied to the bundle adjustment graph could speed the error propagation. A similar problem can be found in heat transfer problems. Say we have a sheet of material with its edges fixed at a lower temperature and at its center heat is applied. Starting with these initial conditions, the temperature of the sheet at steady state can be found using a system of partial differential equations. The surface is discretized into a 2D grid with the temperature given at each point on the grid. An iterative approach such as Gauss-Seidel relaxation can then be used to find the heat distribution. Each iteration propagates the temperature correction a small distance on the grid. This meas that, in a similar way to bundle adjustment error reduction, small scale temperature errors are eliminated quickly while larger scale corrections take many more iterations to propagate through the graph.

The multigrid approach can be used to accelerate the temperature correction propagation and arrive at a solution in fewer, faster iterations. A more complete introduction to the multigrid method can be found in (Briggs et al., 2000) but it will briefly be introduced here. The basic idea of multigrid is to first eliminate the small scale error with one or more iterations. At this point larger scale error corrections will be all that is left. These larger scale errors will take longer to propagate through the graph. However, if we reduce the size of the graph by restriction, say dropping every other grid point we can reduce the problem size by a factor of two (see Figure 5.14). At this coarser scale the large scale error at the fine scale are now smaller scale errors which can be efficiently eliminated. Once the errors

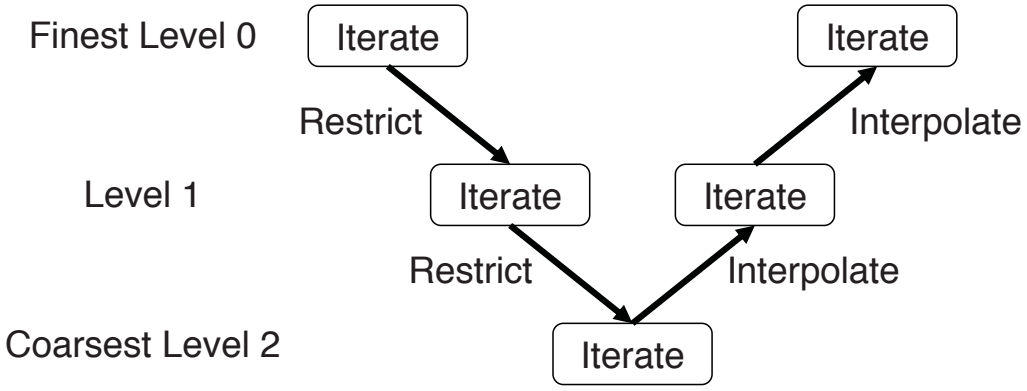

Figure 5.13: A single multi-grid V-cycle.

are corrected at the coarser scale the original grid points are re-inserted by an interpolation operator. This might be as simple as bi-linearly interpolating the temperature values of a coarse grid point's neighbors. Then at the fine level another few iterations are performed to eliminate the error caused by the interpolation. This fine level iteration, restriction, coarse level iteration, interpolation, fine level iteration cycle is referred to as a V-cycle in the multi-grid literature and is shown in Figure 5.13.

Frese and Duckett (2003) showed that a multigrid style approach could be applied to the general SLAM problem. Their approach uses the temporal connections between robot poses as the grid. This temporal sequence can be sub-sampled to arrive at coarser representations of the robot's path. Their approach uses scan matching or odometry to relate one robot pose to the next as well as to relate poses when loops complete. Then each measurement is a transformation with associated uncertainty between two camera poses. When coarsening (restricting) the graph they simply drop every other pose and combine measurements to arrive at measurements with uncertainties linking the dropped pose's neighbors. Converting back from coarse to fine in the upward section of the V-cycle is performed using a linear interpolation of the camera poses at the coarse level.

With a general bundle adjustment problem, the situation is not so simple as Frese and Duckett's case because features are measured in multiple cameras. This means that the

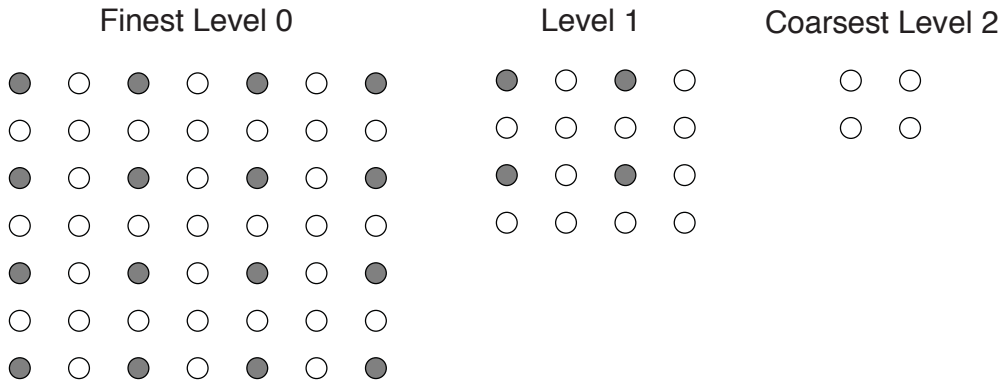

Figure 5.14: The finest level heat distribution solution grid and it's down-sampled counterparts. Grey dots are the grid locations of the heat distribution that are kept on the next coarsest level.

simple inter-robot pose measurement restriction operator they use cannot be applied to the measurements in bundle adjustment. Instead I have developed a bundle adjustment method that restricts the number of cameras in the bundle adjustment while leaving the number of features and measurements the same. Since the 3D features can be eliminated efficiently using the Schur Complement, this restriction operator should allow the error correction to propagate through a bundle adjustment problem more quickly than standard bundle adjustment.

The idea behind this multi-grid approach to bundle adjustment is to adjust not the camera poses themselves, but try to find a temporal error function that describes how the camera pose error accumulates over time. Since the initial camera poses from a VSLAM system come from visual odometry, the error on the poses will be an accumulation of small inter-key-frame errors. This temporal summation of error generates a random walk in 6DOF error space. This random error walk contains trends (generally increasing or decreasing segments) at various temporal scales. This sort of multi-scale error growth can be seen in the one-dimensional random walk in Figure 5.15. This random walk in this figure is the summation of samples from a gaussian distribution with standard deviation one.

Assuming that the initial camera path has error that can be modeled as a random walk,

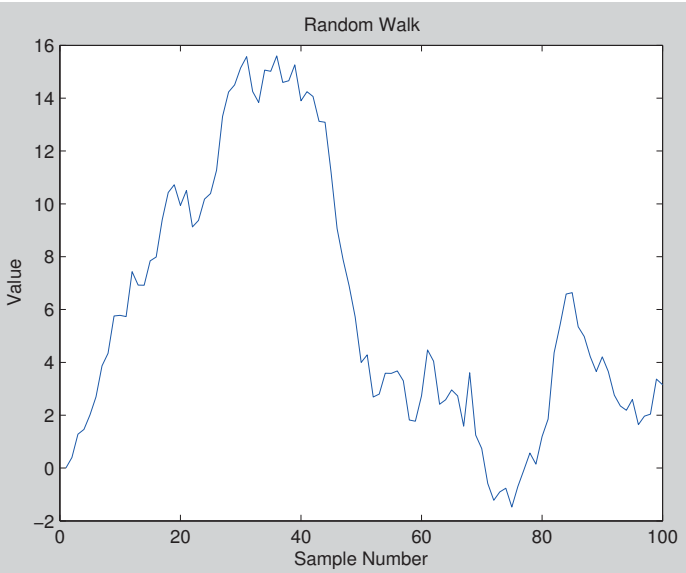

Figure 5.15: A random walk generated by summing one-hundred samples of a normally distributed random variable with standard deviation one. Note the increasing and decreasing trends at various temporal (sample number) scales.

it should be possible to use a sort of multigrid v-cycle to estimate the random error walk's shape. Knowing the error's shape, the inverse error can be applied to the cameras in order to eliminate it and arrive at the minimum of the bundle adjustment function, Equation 5.1. In Figure 5.16 the bottom most pane shows the largest scale error approximation. The two purple cameras are control nodes. The error correction that is applied to the control nodes is linearly interpolated between any two neighboring control nodes in the sequence and applied to the blue cameras in between. The interpolation multiplier is based on the temporal distance between the blue non-control node and its neighboring control nodes in the sequence. The green lines show which error correction is linearly interpolated to find the error correction of a blue camera. In this way in the bottom pane all the cameras can be moved by modifying only the six degrees of freedom of camera seven, not the full thirty-six degrees of freedom of all the cameras (camera zero fixes the gauge). In the middle pane an additional control node is added allowing the error function to be approximated at a smaller, finer scale. In the top pane all cameras are control nodes and the problem is

equivalent to a standard bundle adjustment. Performing a v-cycle, starting at the top pane, moving to the bottom and back up should more efficiently approximate the error function and therefor eliminate the error than standard bundle adjustment because the multi-scale nature of the error is taken into account.

Unfortunately, in practice this has not proven to be the case. A series of experiments was performed comparing standard and multi-grid, error-state bundle adjustment. The performance of both methods tended to scale approximately equally as the number of cameras in a minimization was increased. This may be due to the fact that numerical derivatives were used to find the Jaccobian of the multi-grid BA projection function. This was done because finding the analytic Jaccobians for interpolated error correction values proved to be very complicated, although they may be possible to calculate in the future. A more likely reason why a performance increase was not found is that the measurement space was not down-sampled at the same time as the cameras in the state. In standard multi-grid both the hidden state (the temperature values in a Poisson heat distribution problem) and the measurement error function (the deviation of the temperature values from the values predicted by the Poisson heat equation) are down sampled. However, in my approach the measurements are not down-sampled, all reprojection errors are measured at all scales. A possible way to get around this might be to use synthetic features and measurements in a similar way to the Virtual Key Frame work of Shum et al. (1999). These synthetic features and measurements could summarize the information in a collection of measurements and reduce the measurement space size in addition to the state size.

5.6 Conclusion

In this chapter, we introduced a VSLAM system that fully exploits the recent parallelism gains in consumer computer hardware. The system uses imagery from a stereo camera as its only input. We implemented a two-view consistent 2D tracking module on the GPU

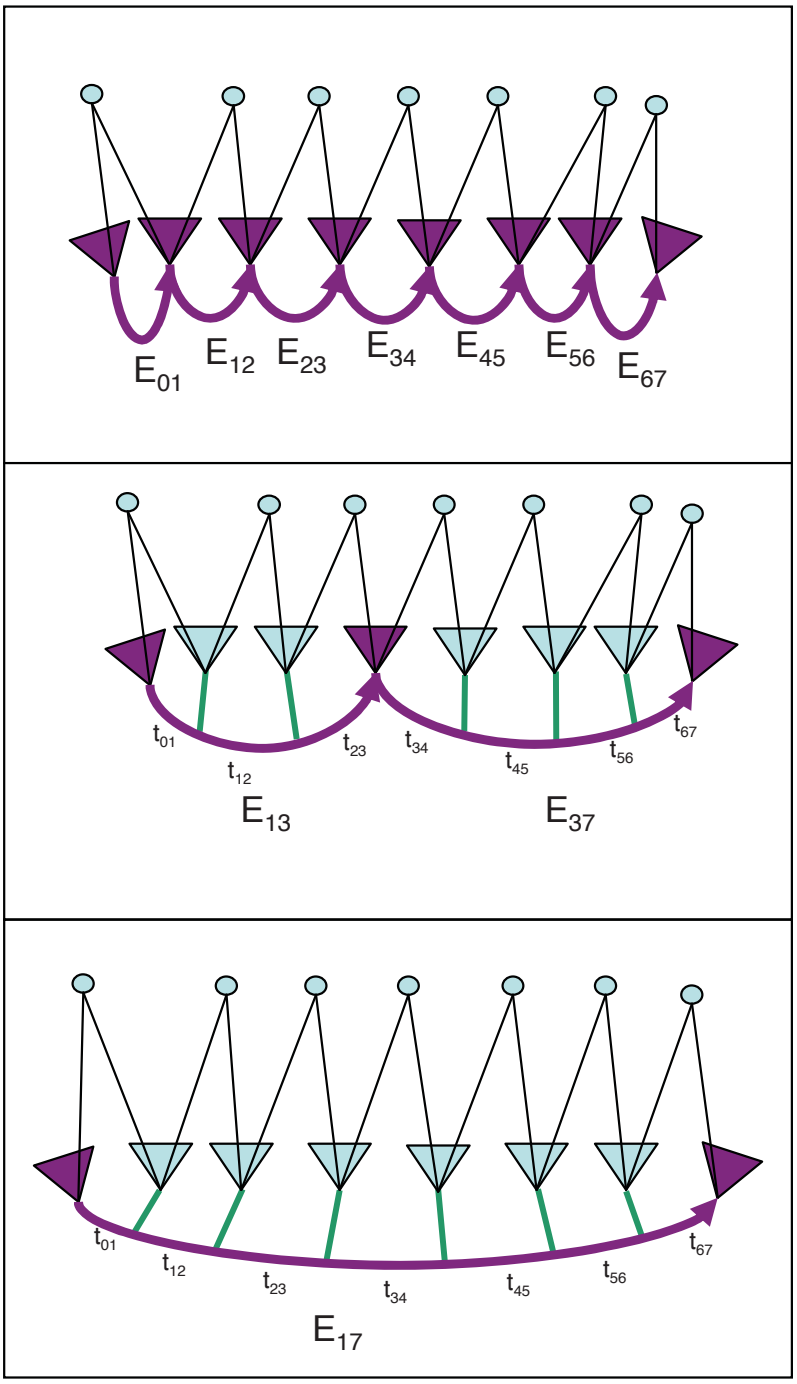

Figure 5.16: Three levels of multigrid, error-state bundle adjustment. The top pane shows the finest scale approximation to the error function, which is equivalent to standard bundle adjustment. The middle and bottom panes show larger temporal-scale approximations to the camera path error. Purple cameras are control nodes which are included in the bundle adjustment problem state. The correction applied to blue cameras is calculated by linearly interpolating the error approximation E_{vw} between the blue camera's two closest control nodes.

to find sparse optical flow. We also used the GPU to extract wide-baseline features for use in loop detection. Our system exploits modern multi-core processors using multiple concurrent threads to perform sparse scene flow, visual odometry and global mapping in parallel. This parallelism allows us to perform the full Euclidean map reconstruction in real time.

While our system operates in real time on office building scale scenes, extension to larger environments remains a challenge. Hierarchical bundle adjustment methods like (Ni et al., 2007a) or (Steedly et al., 2003) may help to overcome these scaling issues. This would allow the system to optimize several sub-problems that are mutually very weakly dependent on each other in parallel. After that, a global combination step could create a globally consistent map.

CHAPTER 6

Conclusion

6.1 Summary

This dissertation has introduced several novel improvements to the problem of multi-camera visual simultaneous localization and mapping. A concise summary of this dissertation's contributions is given in my thesis statment:

- Rigid, two-camera systems with little or no field-of-view overlap can be used in efficient, absolutely scaled visual simultaneous localization and mapping.

- Further, the inherent parallelism in the visual simultaneous localization and mapping problem can be exploited to allow real-time, Euclidean mapping of large areas by decoupling online exploration from loop detection and correction.

In Chapter 3, we developed a solution method for the scaled, six degree of freedom motion of a non-overlapping two-camera system that uses a minimal six temporal feature correspondences. This solution method was studied in detail and degenerate cases were identified where the scale of the motion cannot be calculated, primarily pure translational motion, and motion with the cameras' baseline aligned with the axle of a vehicle in a constant velocity turn.

A second minimal solution method was developed to overcome the inherent degeneracies in non-overlapping stereo cameras while still allowing a wide total field-of-view. The method, presented in Chapter 4, uses a stereo camera with a small region of overlap of the

views. In contrast to the perspective three-point (P3P) method (Haralick et al., 1994), our solution method needs to find only one feature correspondence in the cameras' region of overlap. This is used to find the camera system's translation while the other three degrees of rotational freedom are solved with features in the non-overlapping regions. Our solution method was shown to be more accurate than the P3P method when the cameras' region of overlap is reduced.

Real-time Visual SLAM requires much more than accumulating relative motion estimates to find the camera system's path. It requires real-time feature tracking, visual odometry, loop detection and loop correction. We showed in Chapter 5 that the inherent parallelism in the VSLAM task and a small amount of acceptable latency can be combined to create an effective online, real-time VSLAM system. Three primary computational processes run in parallel in our implementation, scene flow estimation for temporally local correspondence finding, visual odometry for relative ego-motion estimation and global SLAM for loop detection and correction. Splitting the processing into these threads of execution feature tracking can always keep up with the cameras' frame rate and select key-frames based on optical flow for the visual odometry module to process. The visual odometry module can then estimate the relative pose of the cameras using RANSAC while simultaneously extracting wide baseline matchable SIFT features. These relative poses and measurements are then passed to the Global SLAM module which finds loops and corrects the map to reflect them. Current loop correction implementations based on bundle adjustment are too slow to keep up with the rate of key frames. However, our system allows the Global SLAM module to fall behind the visual odometry module while correcting a loop and then later catch up after the loop is completed. By separating visual odometry from loop correction, the camera system can continue to explore its environment while it corrects the map to reflect loops it has already found.

6.2 Future Work

Many interesting, practical problems remain to be solved in visual SLAM. Feature matching over large changes in viewpoint remains a challenge. Using SIFT (Lowe, 2004) or MSER (Matas et al., 2002) features can be reliably matched at up to approximately a thirty degree change in viewing direction. This makes loop detection possible when a camera returns to the same part of a scene from roughly the same direction. However, when entering a hallway from the opposite direction for example, a loop cannot be recognized because features cannot be matched. In the future a variant of the Viewpoint Invariant Patch (VIP) by Wu et al. (Wu et al., 2008) may help with this matching. VIP features can be matched at up to a ninety degree viewing direction difference. While they will make loop completion from greatly varying viewing directions possible they will not allow for the most common case in indoor scenes, returning to the same hallway from the opposite direction.

The VIP is a kind of hybrid between sparse, image based matching methods and dense matching, where a depth value is assigned to most or all of the pixels in an image. With dense matching a more complete geometry for the scene can be recovered. In the future, this more complete geometry may allow loop completion using a form of iterative closest point (ICP) which uses the geometry rather than appearance to join models or detect loops.

Another major problem is extending VSLAM so that it can map larger areas. Promising methods exist such as the out of core bundle adjustment of Ni et al. (Ni et al., 2007a). Their work uses as single level decomposition of the environment to split the bundle adjustment problem into multiple sub-problems. In the future, a multi-level, hierarchical decomposition may allow mapping of areas of virtually unlimited size and complexity (limited by disk space).

Dynamic, changing scenes are another challenge for VSLAM. Current VSLAM approaches assume a static world and try to exclude non-static objects from the map. But over time the environment can change. In offices chairs move, cars enter and leave parking

spaces, buildings are built and torn down, and even the terrain can change through erosion and other processes. Detecting these changes and incorporating them into the map will be necessary for any autonomous system using VSLAM over a long period of time. To know whether a point feature has moved or is simply an outlier it will need to be grouped with other points on a single object or re-detected in the same location over a number of frames. Points on a single object will have to have moved with a single consistent motion to be considered inlier features. If a consistent, non-static motion is found for a group of features, even in a single frame, then the points could be grouped as an object and considered inliers to the camera's motion and the dynamic object's motion with respect to the fixed scene.

One possible approach to dealing with dynamic scenes is to create a sort of four-dimensional map. Static parts of the scene, or parts that have not been detected to change yet, could be placed in one map while a temporal map can be added on top of this. The temporal map would store objects that have changed pose or disappeared with pose information as well as a time of detection. This would allow the system to form a more complete understanding of its environment.

Low power VSLAM for embedded systems such as cell phones also remains a challenge. Klein and Murray's cell phone PTAM (Klein and Murray, 2009) shows that VSLAM can be performed on these limited systems in real-time when the mapped area is very small. However, wider area mapping on these extremely limited systems presents many difficulties. Feature extraction for loop detection requires significant resources, either a high performance processor for SIFT like operators, or a large amount of memory for the classification based feature matching done with Ferns (Özuysal et al., 2010). Loop correction with bundle adjustment is also a compute intensive operation. Map simplification techniques like the ones presented by Klein and Murray (Klein and Murray, 2009) will be needed to reduce the number of variables to optimize and reduce the necessary computation.

A scalable solution to the visual SLAM problem will open up many new possibilities. Robots will be able to navigate unknown areas with low cost, non-emissive sensors, which

do not interfere with each other. Vehicles will be able to drive themselves using sensors that are within the budget of the average consumer. Even now, some vehicle makers are developing stereo vision based systems that will stop the vehicle to prevent crashes and warn of pedestrians stepping into the vehicle's path. The military will be able to deploy small robots to map buildings before soldiers enter them. This will allow the soldiers to get a better impression of the building and its inhabitants, hopefully reducing casualties, both civilian and military, through better situational awareness. This indoor mapping technology will be directly applicable in search and rescue. Robots using VSLAM will be able to explore areas too dangerous for human searchers, discovering survivors that could not otherwise be found. These are just a few of the many ways that VSLAM will change the way we interact with the world.

BIBLIOGRAPHY

Immersive Media Camera Systems Dodeca 2360,http://www.immersivemedia.com/.

Imove GeoView 3000, http://www.imoveinc.com/geoview.php.

Point Grey Research Ladybug2, http://www.ptgrey.com/products/ladybug2/index.asp.

Agarwal, S., Snavely, N., Seitz, S. M., and Szeliski, R. (2010). Bundle adjustment in the large. In *European Conference on Computer Vision*.

Akbarzadeh, A., Frahm, J. M., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Merrell, P., Phelps, M., Sinha, S., Talton, B., Wang, L., Yang, Q., Stewénius, H., Yang, R., Welch, G., Towles, H., Nistér, D., and Pollefeys, M. (2006). Towards urban 3d reconstruction from video. In *Proc. Third International Symposium on 3D Data Processing, Visualization, and Transmission*, pages 1–8.

Amit, Y., August, G., and Geman, D. (1996). Shape quantization and recognition with randomized trees. *Neural Computation*, 9:1545–1588.

Angeli, A., Doncieux, S., arcady Meyer, J., and Ensta, D. F. (2008). Incremental vision-based topological slam. In *in IEEE/RSJ 2008 International Conference on Intelligent Robots and Systems*.

Ankur Handa, Margarita Chli, H. S. and Davison, A. J. (2010). Scalable active matching. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Azarbayejani, A. and Pentland, A. (1995). Recursive estimation of motion, structure, and focal length. *IEEE Transactions on Patttern Analysis and Machine Intelligence*, 17(6):562–575.

Blake, A. and Zisserman, A. (1987). *Visual reconstruction*. MIT Press, Cambridge, MA, USA.

Bolles, R. and Fischler, M. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Bosse, M., Newman, P., Leonard, J., and Teller, S. (2004). Slam in large-scale cyclic environments using the atlas framework. *International Journal of Robotics Research*, 23(12):1113–1139.

Briggs, W. L., Henson, V. E., and McCormick, S. F. (2000). *A multigrid tutorial: second edition*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Brown, M., Hartley, R., and Nistér, D. (2007). Minimal solutions for panoramic stitching. In *IEEE Conference on Computer Vision and Pattern Recognition*, Minneapolis.

Bujnak, M., Kukelova, Z., and Pajdla, T. (2008). A general solution to the P4P problem for camera with unknown focal length. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Byrod, M. and Astrom, K. (2009). Bundle adjustment using conjugate gradients with multiscale preconditioning. In *British Machine Vision Conference*.

Byröd, M., Josephson, K., and Åström, K. (2007). Improving numerical accuracy of Gröbner basis polynomial equation solver. In *IEEE International Conference on Computer Vision*.

Chen, H. and Meer, P. (2003). Robust regression with projection based m-estimators. In *IEEE International Conference on Computer Vision*, volume 2, pages 878–885.

Chli, M. and Davison, A. J. (2008). Active matching. In Forsyth, D., Torr, P., and Zisserman, A., editors, *European Conference on Computer Vision*, volume 5302 of *Lecture Notes in Computer Science*, pages 72–85. Springer.

Chli, M. and Davison, A. J. (2009). Automatically and efficiently inferring the hierarchical structure of visual maps. In *IEEE international conference on Robotics and Automation*, pages 2211–2218, Piscataway, NJ, USA. IEEE Press.

Chow, C. K. and Liu, C. N. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14:462–467.

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

Clipp, B., Frahm, J.-M., Pollefeys, M., Kim, J.-H., and Hartley, R. (2008). Robust 6dof motion estimation for non-overlapping multi-camera systems. In *IEEE Workshop on Applications of Computer Vision*.

Clipp, B., Lim, J., Frahm, J.-M., and Pollefeys, M. (2010). Parallel, real-time visual slam. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Cox, D., Little, J., and O'Shea, D. (1997). *Ideals, Varieties, and Algorithms*. Springer, 2nd. edition.

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.

Davison, A. (2003). Real-time simultaneous localisation and mapping with a single camera. In *IEEE International Conference on Computer Vision*, volume 2, pages 1403–1410.

Davison, A. (2005). Active search for real-time vision. In *IEEE International Conference on Computer Vision*.

Davison, A., Reid, I., Molton, N., and Stasse, O. (2007). Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067.

Dellaert, F., Carlson, J., Ila, V., Ni, K., and Thorpe, C. E. (2010). Subgraph-preconditioned conjugate gradients for large scale slam. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Devernay, F., Mateus, D., and Guilbert, M. (2006). Multi-camera scene flow by tracking 3-d points and surfels. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2203–2212.

Dornaika, F. and Chung, C. (2003). Stereo geometry from 3d ego-motion streams. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 33(2):308 – 323.

Eade, E. and Drummond, T. (2006). Scalable monocular slam. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages I: 469–476.

Eade, E. and Drummond, T. (2007). Monocular slam as a graph of coalesced observations. In *IEEE International Conference on Computer Vision*, pages 1–8.

Eade, E. and Drummond, T. (2008). Unified loop closing and recovery for real time monocular slam. In *British Machine Vision Conference*.

Engels, C., Stewénius, H., and Nistér, D. (2006). Bundle adjustment rules. In *Photogrammetric Computer Vision*.

Fitzgibbon, A. W. and Zisserman, A. (1998). Automatic camera recovery for closed or open image sequences. In *European Conference on Computer Vision*, pages 311–326, London, UK. Springer-Verlag.

Frahm, J. and Pollefeys, M. (2006). Ransac for (quasi-)degenerate data (qdegsac). In *IEEE Conference on Computer Vision and Pattern Recognition*, pages I: 453–460.

Frahm, J.-M., Georgel, P., Gallup, D., Johnson, T., Raguram, R., Wu, C., Jen, Y.-H., Dunn, E., Clipp, B., Lazebnik, S., and Pollefeys, M. (2010). Building rome on a cloudless day. In *European Conference on Computer Vision*.

Frahm, J.-M., Köser, K., and Koch, R. (2004). Pose estimation for multi-camera systems. In *Deutsche Arbeitsgemeinschaft fr Mustererkennung DAGM*.

Frese, U. and Duckett, T. (2003). A multigrid approach for accelerating relaxation-based slam. In *In Proceedings of the IJCAI-03 on Reasoning with Uncertainty in Robotics*, pages 39–46.

Grayson, D. R. and Stillman, M. E. Macaulay 2, a software system for research in algebraic geometry. Available at http://www.math.uiuc.edu/Macaulay2/.

Grossberg, M. D. and Nayar, S. K. (2001). A general imaging model and a method for finding its parameters. In *IEEE International Conference on Computer Vision*, pages 108–115.

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

Haralick, R., Lee, C., Ottenberg, K., and Nolle, M. (1994). Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 13(3):331–356.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151.

Harris, C. G. and Pike, J. M. (1988). 3d positional integration from image sequences. In *Image and Vision Computing*, volume 6, pages 87–90, Newton, MA, USA. Butterworth-Heinemann.

Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition.

Holmes, S. A., Sibley, G., Klein, G., and Murray, D. W. (2009). A relative frame representation for fixed-time bundle adjustment in monocular sfm. In *IEEE International Conference on Robotics and Automation*.

Horn, B. K. P. (1987). Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, 4(4):629–642.

Irschara, A., Zach, C., Frahm, J.-M., and Bischof, H. (2009). From structure-from-motion point clouds to fast location recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2599–2606.

Kaess, M., Ranganathan, A., and Dellaert, F. (2007). isam: Fast incremental smoothing and mapping with efficient data association. In *IEEE International Conference on Robotics and Automation*, pages 1670–1677.

Kim, J., Hartley, R., Frahm, J., and Pollefeys, M. (2007). Visual odometry for non-overlapping views using second-order cone programming. In *Asian Conference on Computer Vision*, pages 353–362.

Kim, J.-H. and Chung, M. J. (2006). Absolute motion and structure from stereo image sequences without stereo correspondence and analysis of degenerate cases. *Pattern Recognition*, 39(9):1649–1661.

Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small ar workspaces. In *IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–10, Washington, DC, USA. IEEE Computer Society.

Klein, G. and Murray, D. (2009). Parallel tracking and mapping on a camera phone. In *Proc. Eigth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'09)*, Orlando.

Konolige, K. and Agrawal, M. (2008). Frameslam: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics*, 24(5):1066–1077.

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

Kukelova, Z., Bujnak, M., and Pajdla, T. (2008). Automatic generator of minimal problem solvers. In *European Conference on Computer Vision*.

Lepetit, V. and Fua, P. (2006). Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1465–1479.

Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathmatics*, II(2):164–168.

Li, H. and Hartley, R. (2006). Five-point motion estimation made easy. In *International Conference on Pattern Recognition*, pages 630–633.

Li, H., Hartley, R., and Kim, J. (2008). A linear approach to motion estimation using generalized camera models. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Lourakis, M. A. and Argyros, A. (2009). Sba: A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software*, 36(1):1–30.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

Lucas, B. D. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679.

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441.

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2002). Robust wide baseline stereo from maximally stable extremal regions. In *British Machine Vision Conference*.

McGlone, C., Mikhail, E., and Bethel, J. (2004). *Manual of Photogrammetry, 5th Edition*. American Society of Photogrammetry and Remote Sensing, Bethesda, MD.

Montiel, J., Civera, J., and Davison, A. (2006). Unified inverse depth parametrization for monocular slam. In *Robotics: Science and Systems*, Philadelphia, USA.

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.

Ni, K. and Dellaert, F. (2006). Stereo tracking and three-point/one-point algorithms - a robust approach in visual odometry. In *International Conference on Image Processing*, pages 2777–2780.

Ni, K., Steedly, D., and Dellaert, F. (2007a). Out-of-core bundle adjustment for large-scale 3d reconstruction. In *IEEE International Conference on Computer Vision*, pages 1–8.

Ni, K., Steedly, D., and Dellaert, F. (2007b). Tectonic sam: exact, out-of-core, submap-based slam. In *IEEE International Conference on Robotics and Automation*, pages 1678–1685.

Nistér, D. (2000). Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors. In *European Conference on Computer Vision*, pages 649–663, London, UK. Springer-Verlag.

Nistér, D. (2003). An efficient solution to the five-point relative pose problem. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages II: 195–202.

Nistér, D. (2003). Preemptive ransac for live structure and motion estimation. In *IEEE International Conference on Computer Vision*, page 199, Washington, DC, USA. IEEE Computer Society.

Nistér, D. (2004). A minimal solution to the generalised 3-point pose problem. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages I: 560–567.

Nistér, D., Naroditsky, O., and Bergen, J. (2004). Visual odometry. volume 01, pages 652–659, Los Alamitos, CA, USA. IEEE Computer Society.

Nistér, D. and Stewénius, H. (2006). Scalable recognition with a vocabulary tree. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2161–2168.

Nistér, D. and Stewénius, H. (2008). Linear time maximally stable extremal regions. In *European Conference on Computer Vision*, pages 183–196, Berlin, Heidelberg. Springer-Verlag.

Özuysal, M., Calonder, M., Lepetit, V., and Fua, P. (2010). Fast keypoint recognition using random ferns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3):448–461.

Özuysal, M., Fua, P., and Lepetit, V. (2007). Fast keypoint recognition in ten lines of code. In *IEEE Conference on Computing Vision and Pattern Recognition*.

Paskin, M. A. (2003). Thin junction tree filters for simultaneous localization and mapping. In Gottlob, G. and Walsh, T., editors, *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*, pages 1157–1164, San Francisco, CA. Morgan Kaufmann Publishers.

Pless, R. (2003). Using many cameras as one. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Raguram, R., Frahm, J.-M., and Pollefeys, M. (2008). A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. In *European Conference on Computer Vision*, pages 500–513, Berlin, Heidelberg. Springer-Verlag.

Scaramuzza, D., Fraundorfer, F., Siegwart, R., and Pollefeys, M. (2009). Absolute scale in structure from motion from a single vehicle mounted camera by exploiting nonholonomic constraints. In *IEEE International Conference on Computer Vision*, pages 1–7.

Schaub, H. and Junkins, J. L. (2003). *Analytical Mechanics of Space Systems*. American Institute of Aeronautics and Astronautics.

Schmid, C. and Zisserman, A. (1997). Automatic line matching across views. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 666 –671.

Shi, J. and Tomasi, C. (1994). Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600.

Shoemake, K. (1985). Animating rotation with quaternion curves. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, New York, NY, USA. ACM.

Shum, H.-Y., Zhang, Z., and Ke, Q. (1999). Efficient bundle adjustment with virtual key frames: A hierarchical approach to multi-frame structure from motion. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Sibley, G. (2009). Relative bundle adjustment. Technical Report 2307/09, Department of Engineering Science, Oxford University.

Sibley, G., Mei, C., Reid, I., and Newman, P. (2009). Adaptive relative bundle adjustment. In *Robotics: Science and Systems*.

Sivic, J. and Zisserman, A. (2003). Video Google: A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision*, volume 2, pages 1470–1477.

Smith, R. C. and Cheeseman, P. (1987). On the representation and estimation of spatial uncertainly. *International Journal of Robotics Research*, 5(4):56–68.

Snavely, N., Seitz, S. M., and Szeliski, R. (2008). Skeletal sets for efficient structure from motion. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Steedly, D., Essa, I., and Delleart, F. (2003). Spectral partitioning for structure from motion. In *IEEE International Conference on Computer Vision*, page 996, Washington, DC, USA. IEEE Computer Society.

Stewénius, H. (2005). *Gröbner Basis Methods for Minimal Problems in Computer Vision*. PhD thesis, Lund University.

Stewénius, H. and Åström, K. (2004). Structure and motion problems for multiple rigidly moving cameras. In *European Conference on Computer Vision*, page 238ff, Prague , Czech Republic. An improved version of this in Chapter 9 of my thesis.

Stewénius, H., Nistér, D., Oskarsson, M., and Åström, K. (2005). Solutions to minimal generalized relative pose problems. In *Workshop on Omnidirectional Vision*, Beijing China.

Tariq, S. and Dellaert, F. (2004). A multi-camera 6-dof pose tracker. In *IEEE and ACM International Symposium on Mixed and Augmented Reality*.

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. The MIT Press.

Tomasi, C. and Kanade, T. (1991). Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University.

Torr, P. H. S. (2002). Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *International Journal of Computer Vision*, 50(1):35–61.

Triggs, B., McLauchlan, P., Hartley, R., and Fitzgibbon, A. (2000). Bundle adjustment – a modern synthesis. In Triggs, B., Zisserman, A., and Szeliski, R., editors, *Vision Algorithms: Theory and Practice*, volume 1883 of *Lecture Notes in Computer Science*, pages 298–372. Springer-Verlag.

Uyttendaele, M., Criminisi, A., Kang, S., Winder, S., Szeliski, R., and Hartley, R. (2004). Image-based interactive exploration of real-world environments. *IEEE Computer Graphics and Applications*, 24(3):52–63.

Wang, J., Zha, H., and Cipolla, R. (2005). Vision-based global localization using a visual vocabulary. In *IEEE International Conference on Robotics and Automation*.

Weng, J. and Huang, T. (1992). Complete structure and motion from two monocular sequences without stereo correspondence. In *Pattern Recognition, 1992. Vol.I. Conference A: Computer Vision and Applications, Proceedings., 11th IAPR International Conference on*, pages 651 –654.

Werner, F., Maire, F., and Sitte, J. (2009). Topological slam using fast vision techniques. In *Proceedings of the FIRA RoboWorld Congress 2009 on Advances in Robotics*, pages 187–196, Berlin, Heidelberg. Springer-Verlag.

Williams, B., Klein, G., and Reid, I. (2007). Real-time slam relocalisation. *IEEE International Conference on Computer Vision*, 0:1–8.

Wu, C. (2007). SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). http://cs.unc.edu/~ccwu/siftgpu.

Wu, C., Clipp, B., Li, X., Frahm, J.-M., and Pollefeys, M. (2008). 3d model matching with viewpoint invariant patches (vips). In *IEEE Conference on Computer Vision and Pattern Recognition*.

Yang, H., Pollefeys, M., Welch, G., michael Frahm, J., and Ilie, A. (2007). Differential camera tracking through linearizing the local appearance manifold. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Zach, C. (2009). GPU KLT tracker http://www.inf.ethz.ch/personal/chzach/oss/GPU-KLT-1.2.zip.

Zach, C., Klopschitz, M., and Pollefeys, M. (2010). Disambiguating visual relations using loop constraints. In *IEEE Conference on Computer Vision and Pattern Recognition*.