Motion estimation from image and inertial measurements

Dennis W. Strelow

November 2004
CMU-CS-04-178

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Dr. Sanjiv Singh, chair
Dr. Martial Hebert
Dr. Michael Lewicki
Dr. Rama Chellappa, University of Maryland

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

This research was sponsored by Exponent, Inc. under a US Army contract, Athena Technologies under US Air Force grant number F08630-03-0024, Allied Aerospace Industries under Army Procurement Office contract number MDA972-01-9-0017, a NASA Graduate Student Researchers Program (GSRP) fellowship under grant number NGT5-50347, and the Honda Motor Company. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: batch shape-from-motion, recursive shape-from-motion, inertial navigation, omnidirectional vision, sensor fusion, long-term motion estimation

Abstract

Robust motion estimation from image measurements would be an enabling technology for Mars rover, micro air vehicle, and search and rescue robot navigation; modeling complex environments from video; and other applications. While algorithms exist for estimating six degree of freedom motion from image measurements, motion from image measurements suffers from inherent problems. These include sensitivity to incorrect or insufficient image feature tracking; sensitivity to camera modeling and calibration errors; and long-term drift in scenarios with missing observations, i.e., where image features enter and leave the field of view.

The integration of image and inertial measurements is an attractive solution to some of these problems. Among other advantages, adding inertial measurements to image-based motion estimation can reduce the sensitivity to incorrect image feature tracking and camera modeling errors. On the other hand, image measurements can be exploited to reduce the drift that results from integrating noisy inertial measurements, and allows the additional unknowns needed to interpret inertial measurements, such as the gravity direction and magnitude, to be estimated.

This work has developed both batch and recursive algorithms for estimating camera motion, sparse scene structure, and other unknowns from image, gyro, and accelerometer measurements. A large suite of experiments uses these algorithms to investigate the accuracy, convergence, and sensitivity of motion from image and inertial measurements. Among other results, these experiments show that the correct sensor motion can be recovered even in some cases where estimates from image or inertial estimates alone are grossly wrong, and explore the relative advantages of image and inertial measurements and of omnidirectional images for motion estimation.

To eliminate gross errors and reduce drift in motion estimates from real image sequences, this work has also developed a new robust image feature tracker that exploits the rigid scene assumption and eliminates the heuristics required by previous trackers for handling large motions, detecting mistracking, and extracting features. A proof of concept system is also presented that exploits this tracker to estimate six degree of freedom motion from long image sequences, and limits drift in the estimates by recognizing previously visited locations.

Acknowledgments

I want to thank my advisor, Sanjiv Singh, for hundreds of hours of research discussions and encouragement; and for listening to, carefully reading, and suggesting improvements in many talks and papers, including this document. I also want to thank Sanjiv for his personal and career advice, which has benefited me very much.

My thanks also go to Martial Hebert, Mike Lewicki, and Rama Chellappa, for agreeing to serve on my committee, and making time for me alongside their students. Their different viewpoints, careful reading, and veteran advice have been very helpful.

I also want to thank Jeff Mishler, for what has now been almost eight years of valuable and direct advice, and for lucidly explaining why gasoline and ammunition are the currency of the future. Similarly, I am indebted to Henele Adams for his expert advice. My thanks go to Tom Minka for immediately reading, and for adeptly finding and correctly a problem in, an earlier version of this work.

Many people helped in capturing the data used in my experiments, including Henele Adams, Dan Bartz, Mark Delouis, Ivan Kirigin, Matt Mason, Jeff Mishler, Wenfan Shi, Jim Teza, Chris Urmson, Michael Wagner, and David Wettergreen, among others.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motion estimation

Robust, long-term motion estimation would be an enabling technology for many problems in autonomous robotics and modeling from video. For example:

- Mars rovers must visit objects of interest that human operators identify in the rover's view. However, the long communications latency between earth and Mars and long periods without communications limit the frequency with which the rover's operators can send commands. So, the rovers are most productive if they can perform these traverses autonomously. Accurate on-board estimation of the rover's motion is an important component in performing these traverses successfully.

- Astronauts on the International Space Station sometimes perform dozens of experiments a day, spanning many disciplines. NASA Ames Research Center has been developing a softball-sized Personal Satellite Assistant (PSA)[4][3] that can move around the station in six degrees of freedom to maximize the astronaut's productivity during these experiments and monitor conditions on the station. For example, the PSA would allow experts on earth to move around the station and collaborate in experiments virtually, via the PSA's cameras; would provide a hands-free terminal for astronauts performing delicate experiments; and would monitor conditions in confined locations inaccessible to the astronauts. Each of these capabilities requires that the PSA be able to estimate its position on the station, accurately and over the long-term.

- Micro air vehicles (MAVs) are under development that can be launched from the ground, and carry a video camera and chemical sensors for surveillance, search and rescue, air sampling, and other applications. AeroVironment's Black Widow[24], for instance, has a wingspan of 6 inches, a mass of less than 100 grams, a maximum communications range of 2 kilometers, and an endurance of 30 minutes, making it a good candidate for discrete surveillance over the horizon, around the corner in the urban canyon, or under the canopy in wooded areas. However, to report the location of activity or chemical threats, the vehicle's position must be accurately estimated.

- Recent search and rescue robot designs are capable of moving over highly uneven terrain, into spaces deep within rubble. To report the location of survivors, these robots should estimate their position in six degrees of freedom.

- Modeling complex environments like building interiors and cities from image sequences is a long-standing problem in computer vision research (for example, see El-Hakim[13]). If the camera's motion can be accurately established, then the problem reduces to modeling the geometry and surface properties of objects in the environment from images taken at known locations.

Each of the following is required by some or all of these example problems.

- **Six degree of freedom motion.** Motion estimation for the Personal Satellite Assistant, micro air vehicles, and modeling complex environments are inherently six degree of freedom problems. Mars rovers and search and rescue robots are ground vehicles but may move over nonplanar terrain, so their motion is best modeled using six degrees of freedom.

- **Motion in unknown environments.** Mars rovers, micro air vehicles, search and rescue robots, and modeling complex environments must all operate in unknown environments. While the International Space Station is a partially known environment, it is subject to change as components are moved around the station, and known fiducials are not welcome on the station.

- **Motion over the long term.** All of the examples require estimation over the long term.

- **Motion from small, light, and inexpensive sensors.** The Personal Satellite Assistant requires small and light sensors. Micro air vehicles require small, light, and *cheap* sensors, since ideal micro air vehicles are disposable.

- **Motion without GPS or other absolute position systems.** GPS is not available on Mars or on the International Space Station, and other systems that might be used for absolute positioning, such as known fiducials or beacons, are not suitable for use on the station. Micro air vehicles and search and rescue robots may enter environments, such as building interiors or rubble, respectively, where GPS cannot be received; and MAVs' access to GPS can be jammed. While systems for modeling complex environments can use GPS outdoors, GPS is not useful for modeling interiors.

Many common sensors are poor candidates given these requirements. Odometry is not suitable for six degree of freedom estimation. Range devices are often large, heavy, and expensive; and are limited in range. Inertial sensors alone are not suitable for motion estimation over the long-term, because integrating their measurements produces drift in the estimated motion.

However, cameras are good candidates for each of these problems. Cameras can be used to estimate six degree of freedom motion in unknown environments, without GPS and over the long term, as described in the next section. Cameras can be made small, light and cheap. Furthermore, cameras are not inherently limited in range and do not emit any energy into the environment.

1.2 Motion estimation from image measurements

As mentioned in the previous section, cameras are good candidates for six degree of freedom motion estimation, and many methods exist for extracting motion from images. One paradigm splits the problem into two steps:

1. Extract and track sparse image features through the image sequence.

2. Estimate the six degree of freedom camera position at the time of each image.

The first step, feature extraction and tracking, establishes the image locations ("projections") of points in the scene from different vantage points. This step is normally performed with one of two approaches, called *template matching* and *feature extraction and matching* in subsequent sections. Sections 2.5 and 2.6 below discuss sparse feature tracking and these two approaches, respectively, in more detail. Chapter 6 describes a new tracker that, among other qualities, combines the best of these two approaches.

Many algorithms exist for the second step, estimation, with different advantages and disadvantages. Some existing algorithms for the estimation step figure prominently in the following chapters:

1. Bundle adjustment[61][57] uses general nonlinear minimization to simultaneously find the six degree of freedom camera position at the time of each image, and the

three-dimensional point positions, that maximize the likelihood of the observed tracking data. Bundle adjustment is a batch method, meaning that all of the observations are required before the computation begins; is iterative, so an initial estimate of the camera and point positions are required; and can be computationally expensive. However, bundle adjustment naturally accommodates arbitrary camera models and missing observations (i.e., the case where features enter and leave the field of view), and is usually considered the gold standard algorithm. Bundle adjustment is reviewed below in Section 2.10.

2. The iterated extended Kalman filter (IEKF)[16] can be used to estimate the same unknowns and minimize roughly the same function as bundle adjustment[5], but in a recursive fashion that processes tracked features from each image as they arrive. Motion estimation using Kalman filtering can be performed in real-time, but as Chapters 4 and 5 below explain, Kalman filtering for motion estimation is problematic in the case of missing observations, and suffers from poor a priori assumptions on the camera motion.

3. The variable state dimension filter (VSDF)[38] is a recursive method that combines aspects of bundle adjustment and Kalman filtering. Like Kalman filtering algorithms for motion estimation, the VSDF estimates the same unknowns and minimizes roughly the same error function as bundle adjustment. However, the VSDF is explicitly designed to handle the case of missing observations within a recursive framework, and jettisons the assumptions on camera motion that hamper Kalman filter algorithms. On the other hand, the VSDF has difficulties handling features leaving and then reentering the image sequence.

4. Two- and three-frame methods for shape-from-motion solve for the camera motion between pairs and triples, respectively, of images in the sequence, and concatenate the two- and three-frame motions to produce an estimate of the overall ("multi-frame") motion[26]. These methods can be efficient, but they are ad hoc in that they do not produce or approximate the optimal estimate of the overall motion. The tracker described in Chapter 6 below uses two- and three-frame estimation in the context of sparse image feature tracking.

Algorithms 1, 2, and 3 are shape-from-motion algorithms, meaning that they simultaneously estimate both shape (the three-dimensional point positions) and motion (the six degree of freedom camera positions). The two- and three-frame methods are also sometimes referred to as shape-from-motion algorithms, even though the two- and three-frame motions can be found without simultaneously finding the shape.

The algorithms in subsequent chapters are based on these shape-from-motion algorithms, but in this work the term “shape-from-motion” is usually deprecated in favor of “motion estimation” for two reasons. First, “shape-from-motion” implies that the goal is to recover shape, with the motion recovered as a necessary intermediate quantity, whereas in our applications the situation is exactly the opposite. Second, “shape-from-motion” refers to algorithms that recover motion from image measurements, whereas much of the emphasis in subsequent chapters is on motion from both image and inertial measurements.

In addition to the problems listed above for each estimation algorithm, motion estimation from images suffers from some long-standing difficulties. Estimation generally suffers from:

- sensitivity to incorrect or insufficient image feature tracking
- sensitivity to camera modeling and calibration errors
- sensitivity to outlier detection thresholds
- sensitivity to degenerate camera motions
- long-term drift in scenarios with missing observations, due to all of the above
- ambiguities, such as depth reversal, in the shape and motion estimates

In addition, batch algorithms such as bundle adjustment suffer from:

- the need for an initial estimate, and poor convergence or failure to converge if the initial estimate is poor
- poor convergence or failure to converge if the percentage of missing observations is high

and recursive algorithms suffer from:

- poor approximations in modeling the state prior distribution
- poor assumptions about the camera motion, e.g., smoothness

1.3 Robust motion estimation from image measurements

This thesis investigates some strategically chosen approaches to some of these problems.

1.3.1 Motion estimation from omnidirectional image measurements

Motion from omnidirectional images is a promising approach for removing many of the sensitivities that cause gross location errors in motion estimates, including sensitivity to incorrect feature tracking, sensitivity to outlier thresholds, and ambiguities. In addition, because features may be visible through more of an omnidirectional image sequence, omnidirectional images may significantly reduce drift in long image sequences.

Chapter 3 below explores motion estimation from omnidirectional images. An omnidirectional projection model is developed for use with existing motion estimation algorithms such as bundle adjustment and Kalman filtering, which accommodates the wide design space of noncentral omnidirectional cameras and misalignment between the camera and mirror. A calibration algorithm is described for determining the misalignment, and the effect of omnidirectional images and this precise calibration on motion estimation accuracy are examined.

1.3.2 Motion estimation from image and inertial measurements

Image and inertial measurements are highly complimentary modalities for motion estimation. As mentioned in Section 1.1, inertial measurements alone are not appropriate for motion estimation, because random errors, unmodeled nonlinearities, and slowly changing biases in inertial measurements cause the motion estimates to drift. With image measurements, however, this drift can be minimized and other unknowns needed to interpret the inertial measurements, such as the gravity direction and magnitude, can be estimated. On the other hand, inertial measurements can help to reduce the sensitivity of image-only motion estimation to feature tracking errors and camera calibration errors, and establish the global scale of the motion, which cannot be recovered from image measurements alone.

Chapter 4 describes both batch and recursive algorithms for estimating sensor motion, sparse scene structure, and other unknowns from image, gyro, and accelerometer measurements. These algorithms are designed to recover accurate estimates of the unknowns, if possible, even when the estimates from image or inertial measurements alone may be grossly wrong. Chapter 5 uses these algorithms and a large suite of experiments to explore the relative sensitivity of image-only and image-and-inertial motion estimation, the relative advantages of image-and-inertial estimation and estimation from omnidirectional images, and many other issues.

1.3.3 Constrained image feature tracking

As mentioned above, motion estimation can be sensitive to incorrect or insufficient feature tracking. Poor tracking can result from:

- large image motions
- unmodeled changes in features' appearance between images, resulting from lighting changes, specularities, occlusions, or foreshortening
- insufficient, repetitive, or one-dimensional image texture
- poor spatial distribution of features in the image

The long-time go-to feature tracker in the shape-from-motion community, Lucas-Kanade, is susceptible to all of these problems. The work presented in this thesis has developed a new tracker that exploits SIFT keypoints[36] to establish the epipolar geometry between images, constrains tracking between images to one dimension, and eliminates Lucas-Kanade's heuristics for handling large motions, detecting mistracking, and extracting features. The resulting tracker qualitatively outperforms Lucas-Kanade on many real sequences, and is described in detail in Chapter 6, along with experimental results.

1.3.4 Long-term motion estimation

Motion estimates from image sequences with high percentages of missing observations, such as those taken from a moving vehicle, contain drift. While this drift can be reduced by carefully addressing the issues described in Section 1.2 above, drift cannot be eliminated completely because the sensors' position is not being estimated relative to a single, always visible external landmark.

If the camera repeatedly moves between locations in a bounded area, however, it should be possible to limit drift by:

1. recognizing when a previously mapped area has been revisited
2. using new observations of the revisited area to limit drift in the estimates

While the second problem has been widely explored in the simultaneous localization and mapping literature in the context of ground rovers with range scanners and odometry, the first problem is unique to image-based motion estimation. Chapter 7 describes a proof of concept system that exploits the tracker described in Chapter 6, the variable state dimension filter, and SIFT keypoints to demonstrate limited drift.

1.4 Organization

The organization of this document is as follows. Chapter 2 reviews related work and presents the background material that is common to the subsequent chapters. Chapters 3-7 investigate the avenues of approach described in Section 1.3 above:

- Chapter 3 describes motion estimation from omnidirectional images.
- Chapters 4 and 5 describe algorithms and experimental results for motion estimation from image and inertial measurements, respectively.
- Chapter 6 describes the new image feature tracker.
- Chapter 7 describes long-term motion estimation.

Chapter 8 concludes by recapping the conclusions and contributions of the work, giving some explicit recommendations, and describing possible future directions.

Chapter 2

Related work and background

As described in the introduction, this work investigates a number of approaches to improving motion estimation from image measurements. Sections 2.1-2.8 in this chapter review previous work related to these approaches. In addition, Sections 2.9 and 2.10 review conventional camera projection models and bundle adjustment, respectively, which are used extensively in subsequent chapters.

2.1 Omnidirectional vision

Recent omnidirectional camera designs include catadioptric designs, which combine a conventional camera with a mirror that expands the camera's field of view. During the last few years, researchers have been active in the design, modeling, and calibration of these cameras, and in exploiting these cameras for motion estimation and localization. As the overview in Chapter 3 explains, the wide field of view provided by these cameras is promising for improving the robustness and accuracy motion estimation from images.

The recent widespread interest in catadioptric cameras in the computer vision community was sparked by Nayar's design[41], which combines an orthographic camera with a parabolic mirror. This design achieves a single viewpoint, i.e., rays that are reflected from the mirror into the conventional camera's center also intersect at a single point inside the mirror. This single viewpoint property allows portions of the omnidirectional image to be correctly remapped to perspective views, and the viewpoint inside the mirror can be adopted as the camera's center for the purpose of defining the camera's epipolar geometry. The entire class of single viewpoint catadioptric designs was explored by Baker and Nayar[1].

The definitive study of single viewpoint cameras for three-dimensional computer vision is given in a sequence of papers by Geyer and Daniilidis, who emphasize Nayar's design. Their investigation includes modeling[17], epipolar geometry[22], calibration[20][17], rectification[21], projective geometry[18], and uncalibrated structure and motion[19] for these cameras.

Other work on single viewpoint camera calibration includes Kang[35], who describes two methods for single viewpoint camera calibration. The first recovers the image center and mirror parabolic parameter from the image of the mirror's circular boundary in one image. The second method uses minimization to recover skew in addition to the other calibration parameters. Other work on motion estimation with single viewpoint cameras includes Gluckman and Nayer[23], who extend three algorithms for ego-motion estimation with conventional cameras to the single viewpoint case.

On the other hand, relatively little work has been done with non-single-viewpoint, or noncentral, catadioptric cameras. Examples of noncentral catadioptric cameras include Chahl and Srinivasan's design[8], which sacrifices the single viewpoint property to gain nearly uniform image resolution between the omnidirectional image's center and periphery. Ollis' design[46], which is used in this work, refines Chahl and Srinivasan's design to achieve exactly uniform resolution between the image's center and periphery. Single viewpoint cameras where there is a known misalignment between the camera and mirror lose their single viewpoint property, but can be modeled as noncentral cameras.

Chapter 3 below presents an omnidirectional camera model that accommodates noncentral cameras and known misalignment between the camera and mirror; camera calibration using this model; and camera motion estimation using this model. Chapter 5 investigates the relative advantages of motion estimation from omnidirectional cameras and from conventional camera and inertial measurements.

2.2 Batch motion estimation from image and inertial measurements

Chapter 4 below describes batch and recursive methods for estimating sensor motion, sparse scene structure, and other unknowns from image, gyro, and accelerometer measurements. This section and the next briefly review the existing methods for estimating sensor motion from image and inertial measurements that are most closely related to these methods. While the majority of existing methods are recursive, a few relevant batch methods do exist, and both batch and recursive methods for this problem are reviewed, in this section and in Section 2.3, respectively.

Deans and Hebert[11] describe a batch method for bearings-only localization and mapping that uses Levenberg-Marquardt to estimate the planar motion of a vehicle and the two-dimensional location of landmarks observed by the vehicle's omnidirectional camera, from the landmarks' vehicle coordinate system bearings (one-dimensional projections) and the vehicle's odometry. The error function described in Section 4.3, which utilizes image, gyro, and accelerometer measurements, is closely related to Deans and Herbert's image-and-odometry error function. However, estimating six degree of freedom motion from image measurements and inertial sensors introduces some difficulties that do not arise in estimating planar motion from bearings and odometry. In particular, using image measurements for six degree of freedom motion normally requires careful modeling and calibration of the camera, especially in the omnidirectional case, whereas this modeling is not required in two dimensions. In addition, the use of accelerometer observations for six degree of freedom motion requires estimation of the vehicle's velocity and orientation relative to gravity, which odometry does not require. In subsequent work[12], Deans also considered iteratively reweighted least squares (IRLS) for robust estimation within the batch framework, to improve the quality of estimates in the presence of image feature mistracking and other gross observation errors.

A second batch method is described by Jung and Taylor[33]. This method applies shape-from-motion to a set of widely spaced keyframes from an omnidirectional image sequence, then interpolates the keyframe positions by a spline that best matches the inertial observations. The resulting algorithm provides a continuous estimate of the sensor motion, and only requires that feature correspondences be established between the keyframes, rather than between every image in the sequence. Since the image and inertial measurements are not used simultaneously, however, the interpolation phase will propagate rather than fix errors in the motion estimated in the shape-from-motion phase.

2.3 Recursive motion estimation from image and inertial measurements

Huster and Rock[30][31][29][32] detail the development of a recursive algorithm for estimating the six degree of freedom motion of an autonomous underwater vehicle (AUV) using gyro measurements, accelerometer measurements, and the image measurements of a single point in the vehicle's environment. In Huster and Rock[30], the authors develop an extended Kalman filter (EKF) and a two-step algorithm for a simplified, two-dimensional version of the motion estimation problem. The difficulties with linearizing the measurement equations about uncertain estimates in the EKF are sidestepped in the two-step filter, which chooses the state so that the image and inertial measurement equations become linear, and avoids linearization in the state time propagation using the unscented filter. In Huster and Rock[29], a full three-dimensional version of the two-step filter is constructed, and integrated with a controller and a method for choosing an endpoint trajectory that optimizes the quality of the motion estimates. The authors' experiments show that the resulting system reliably performs a grasping task on a manipulator. In the measurements employed and in the quantities estimated, this system has many similarities to the iterated extended Kalman filter (IEKF) for motion estimation described in Section 4.4, but is more sophisticated in its handling of nonlinearities. However, the use of a single image point feature, which is motivated by the potentially poor quality of underwater images, precludes the use of Huster and Rock's method for long distance traverses.

You and Neumann[62] describe an augmented reality system for estimating a user's view relative to known fiducials, using gyro and image measurements. This method is simpler than Huster and Rock's in that it does not employ an accelerometer, which is a more difficult instrument to incorporate than a rate gyro, but expands the scene from a single point to a set of known points. Rehbinder and Ghosh[52] also describe a system for estimating motion relative to a known scene, in this case containing three-dimensional lines rather than point features. Rehbinder and Ghosh incorporate accelerometer measurements as well as gyro and image measurements.

Qian, *et al.*[50] describe an extended Kalman filter (EKF) for simultaneously estimating the motion of a sensor rig and the sparse structure of the environment in which the rig moves, from gyro and image measurements. The authors show motion estimation benefits from the addition of gyro measurements in several scenarios, including sequences with mistracking and "mixed domain" sequences containing both sensor translation and pure rotation. This system is more general than that described by Huster and Rock, You and Neumann, or Rehbinder and Ghosh, in that the sparse scene structure is estimated in addition to the motion rather than known, but this system makes the implicit assumption that each scene point is visible in every image of the sequence. In later work, Qian and Chellappa[49] also investigated motion estimation from image and gyro measurements within a sequential Monte Carlo framework. In this case, the authors showed that the inclusion of gyro measurements significantly reduced the number of samples required for accurate motion estimation.

Chai, *et al.*[9] describe a system for simultaneously estimating the motion of a sensor rig and the sparse structure of the environment in which the rig moves, from gyro, accelerometer, and image measurements. This system estimates nearly the same unknowns as the algorithm described in Chapter 4, but divides the estimation task between a motion filter, which estimates motion by assuming that the scene structure is known, and a structure filter, which estimates the scene structure by assuming the motion is known. The two filters are combined into a system for simultaneous estimation of motion and scene structure by supplying the estimates from each filter as the known inputs to the other. While this arrangement improves efficiency, it will result in artificially low covariances on the estimated motion and structure, particularly for long-term problems where due to drift, the motion and the scene structure are likely to contain large but coupled errors. The authors do not explicitly consider the problem of estimating motion in sequences where points enter and leave the image sequence. The authors also consider the relative benefits of using one and two cameras in synthetic tests, and conclude that the use of two cameras can produce estimates with significantly lower errors.

The system described by Mukai and Ohnishi[40] also simultaneously estimates the motion of a sensor rig and the sparse structure of the environment in which the rig moves using gyro and image measurements. In Mukai and Ohnishi's method, the motion between pairs of images is estimated up to a scale factor, and the estimated motion is used to determine the structure of the points seen in both images. These pairwise estimates are then merged sequentially by applying the scaled rigid transformation that best aligns the recovered structures. This method handles sequences where points do not appear in every image, but both the pairwise motion recovery and merging steps of this method are ad hoc. For instance, this method does not maintain any measure of the error in the resulting motion estimates.

Foxlin[15] describes a general framework for recursive simultaneous localization, mapping, and sensor calibration. The system consists of a decentralized filter that integrates the results of three complimentary error filters for localization only, localization and mapping, and localization and sensor calibration. This architecture reduces the computation relative to including environmental object locations and sensor calibration parameters in the same state vector, and allows mapping and sensor calibration to be easily removed from the estimation once the positions of environmental objects or sensor bias values have been estimated with sufficient accuracy. In [14], the authors describe one instance of this architecture, which uses gyro measurements, accelerometer measurements, and the image measurements of fiducials whose appearance but not (in general) location are known. In the case of auto-mapping, where the locations of fiducials are not known, the system pose is first determined relative to four fiducials whose x, y, z positions are known, and the positions of subsequently acquired fiducials are entered into the filter using the image location and distance to the fiducial estimated from the fiducial's image size. In the system described here, the initial position is initialized without *a priori* knowledge of any feature locations using a batch algorithm, and the positions of natural features are initialized using triangulation from multiple image positions, which is more appropriate for features whose appearance is not known *a priori*. While the authors report real-time performance for an initial implementation of automapping, they do not report on the accuracy of their system for automapping.

In addition to the batch algorithm described in Section 2.2, Deans[12] describes a hybrid batch-recursive method that estimates the planar motion of a vehicle and the two-dimensional location of landmarks observed by the vehicle's omnidirectional camera from the landmarks' vehicle coordinate system bearings (one-dimensional projections) and the vehicle's odometry. This method adapts the variable state dimension filter (VSDF), originally described by McLauchlan[38] for shape-from-motion, to recursively minimize the same image-and-odometry error function minimized by Deans and Hebert's batch method. This approach naturally handles cases where points enter and leave the image sequence, and delays the linearization of measurements until the estimates used in the linearization are more certain, reducing bias in the state estimate prior. A similar adaptation of the batch algorithm for estimating motion from image and inertial measurements described here, using the VSDF, would be a natural extension of this work.

2.4 SLAM and shape-from-motion

Simultaneous localization and mapping (SLAM) and shape-from-motion (SFM) are two broad areas concerned with simultaneously estimating sensor motion and scene structure. This section briefly contrasts these problems and discusses their relevance for long-term image-based motion estimation. This section does not describe specific methods for SLAM and shape-from-motion. Instead, see Deans[12] for a good overview of specific methods for these problems.

Algorithms for simultaneous localization and mapping (SLAM) typically estimate planar vehicle motion and the two-dimensional positions of landmarks in the vehicle's environment using observations from the vehicle's odometry and from a device that provides both the range and bearing to the landmarks. Since both range and bearing are available, the device provides noisy two-dimensional landmark positions in the device's coordinate system, and the major technical problem becomes the correct incorporation of these vehicle system measurements into the global coordinate system mean and covariance estimates. Recent work on SLAM focuses on mapping large areas, in which each landmark may only be visible from a small portion of the vehicle path, in a recursive fashion; and on "closing the loop", which exploits a landmark that has been lost and reacquired to maintain the topological consistency of the reconstructed motion and landmark positions.

Algorithms for shape-from-motion typically estimate the six degree of freedom motion of a camera and the three-dimensional position of points, from point features tracked in the camera's image sequence. Here, the observations the camera provides are the two-dimensional projections of landmark positions in the camera's coordinate system, rather than the three-dimensional camera system position, so in SLAM terminology, one would say that the camera provides bearings but not range. So, shape-from-motion typically estimates more unknowns than SLAM from less generous data. However, very little work has been done on automatically mapping large areas or on closing the loop using recursive shape-from-motion.

The motion estimation algorithms in Chapter 5 estimate six degree of freedom motion, sparse scene structure, and other unknowns from image, gyro, and accelerometer measurements. So, they fall somewhere between SLAM and shape-from-motion in terms of the observations and recovered unknowns. In terms of the technical approach, the algorithms described in Chapter 5 are more akin to existing algorithms for SFM than to approaches for SLAM. This bias toward shape-from-motion is also reflected in the work on long-term motion estimation in Chapter 7, and future algorithms would benefit from SLAM approaches to estimating errors and closing the loop within a recursive framework.

2.5 Data association and image feature tracking

In many scenarios where multiple targets are tracked over time using radar, sonar, or similar sensors, the correspondence between the targets and the measurements they generate at any one time is required for tracking the targets, but is not given explicitly. The problem of matching measurements with the tracked targets that generated them is called data association. Because the measurements alone may provide no information about which target generated them, methods for data association typically associate measurements with a target based on the likelihood of the measurements given the target's estimated kinematics. So, the problems of target tracking and data association become coupled. In scenarios where the targets may cross each other, may maneuver erratically, and may not produce a measurement at each time, elaborate data associate algorithms may be necessary. A good review of these issues and the corresponding approaches for tracking and data association is given by Bar-Shalom and Li[2].

The data association paradigm has been widely adopted in SLAM for associating sonar or laser returns with the landmarks in the environment that generated them. Montemerlo and Thrun's approach[39] is a recent example. In the computer vision community, the data association paradigm has been investigated for image feature tracking by Rasmussen and Hager[51]. In this case, the authors applied two baseline data association methods, the probabilistic data association filter (PDAF) and joint probabilistic data association filter (JPDAF), to the problems of tracking homogeneous colored regions, textured regions, and nonrigid image contours.

In shape-from-motion and image-based motion estimation, the targets and measurements are three-dimensional points and their image projections, respectively. There are two major differences between this scenario and traditional data association applications like associating radar returns with the target that generated them. First, the image appearance in the neighborhood of each projection provides strong identifying information about the tracked point. Second, the scene is assumed rigid, so that targets are not moving independently. As a result, there is less motivation for using kinematics in image feature tracking than in traditional data association problems.

There are two variations on the image feature tracking problem. In the first, a large number of features, typically 50-200, must be tracked between adjacent images in video so that the camera motion relative to recent times can be estimated. For this problem, template matching or feature extraction and matching have often been used in the shape-from-motion literature, and these techniques are briefly reviewed in Section 2.6 below.

In the second variation, features that have left the camera's view and later reentered, possibly at a much later time, should be recognized so that the camera's position relative to a more distant time can be estimated. Using the estimated camera and point positions for this purpose is promising, and would couple the estimation and matching just as tracking and data association are coupled in other applications. Possible approaches for this problem are described in Section 2.8.

2.6 Sparse image feature tracking

Sparse feature tracking provides the image positions of three-dimensional points from different vantage points that make shape-from-motion and motion estimation from images possible. Two paradigms for sparse feature tracking are common.

Template matching is the first paradigm. In this approach, a small intensity template is extracted around the feature's location in the current image, and the feature's location in the new image is found by minimizing the difference between the template intensities and the intensities around the location in the new image. The template sizes, the difference metric, and the parameterization of the features' locations in the new image vary. For example, the affine transformation of a 21×21 pixel template might be found that minimizes the sum-of-absolute-differences (SAD) between the template intensities and the new image intensities.

The methods used to search for the minimizing location in the new image also vary. A brute force approach computes the difference metric at discrete locations in the new image, within some rectangle around the feature's location from the previous image. Lucas-Kanade is a more elegant approach, which uses iterative optimization to minimize the sum-of-squared-differences (SSD) between the template and new image intensities. Lucas-Kanade has long been the go-to tracker in the shape-from-motion community, but is not sufficient for demanding applications. Lucas-Kanade, its advantages, and its shortcomings for motion estimation are described in detail in Sections 6.2 and 6.3.

Independent extraction and matching is the second paradigm. In this approach, salient features are extracted independently in the two images, and a consistent set of matches between the two sets of features is found using a combinatorial search. Matched features are then linked across multiple images to produce feature tracking across multiple views. Systems adopting this approach include DROID[25], which extracts Harris features in each image, and matches features between images based on the intensities in the features' neighborhoods, geometry consistency, and the estimated camera kinematics. Nistér, *et al.*[45] describe a real-time system for camera motion estimation that also extracts Harris corners and matches them based on nearby intensities and geometric consistency. This algorithm is described in more detail in Section 2.7 below, in the context of shape-from-motion with missing observations.

A disadvantage of independent extraction and matching is that it only tracks features as long as they are extracted in the new image. So, the average length of feature tracks is limited by the repeatability of extraction in the presence of feature appearance changes. For example, Harris reports roughly 80% repeatability for the Harris extractor[25]. So, we would expect to see a feature in only several consecutive images before it is lost.

SIFT keypoints[36] are a recent development that can also be used for feature extraction and matching. SIFT keypoints are image locations that can reliably extracted to subpixel resolution under image transformations such as rotation, scaling, and affine or projective transformations. A SIFT feature vector associated with each keypoint describes the image intensities near the keypoint in a way that is also nearly invariant to these transformations, providing a simple method for matching corresponding keypoints in two images without assumptions on the relative position (e.g., proximity) of the keypoints in the two images.

Chapter 6 below describes a new tracker that, among other qualities, combines these paradigms. SIFT keypoints and their associated feature vectors are first used for feature extraction and matching, respectively. These matches are used to establish the epipolar geometry between adjacent images in the sequence. Discrete template matching and one-dimensional Lucas-Kanade are then used to find the rough position for features of interest on their epipolar lines in the new image, and to refine those positions, respectively. The tracker's use of independent extraction and matching and of template matching, are described in detail in subsections 6.5.2 and 6.5.3, respectively.

2.7 Shape-from-motion with missing observations

Shape-from-motion problems with missing observations, in which features become occluded, enter and leave the field of view, or are lost due to mistracking, are a long-standing difficulty in shape-from-motion. Among batch algorithms for shape-from-motion, bundle adjustment[61][55] and iterative versions[47] of the factorization method accommodate missing observations. Among recursive algorithms, the variable state dimension filter (VSDF) is explicitly designed to handle missing observations, and Section 4.4 below considers missing observations in the iterated extended Kalman filter (IEKF) framework. Ad hoc algorithms that operate by sequentially concatenating 2-frame and 3-frame shape-from-motion solutions, which are discussed in more detail at the end of this section, naturally handle missing observations.

However, three issues remain:

1. If the percentage of missing observations is high, bundle adjustment and iterative factorization algorithms will fail to converge, or converge much too slowly.

2. In features rapidly enter and leave the field of view, the camera's position can no longer be estimated with respect to a single, always visible external landmark. So, even small errors in feature tracking and camera intrinsics calibration necessarily lead to drift in the camera position estimates.

3. Features that leave and then reenter the field of view are problematic for recursive methods, even in the case of the VSDF, which is explicitly designed to handle missing observations.

Poelman's[47] exploration of his iterative factorization algorithm's convergence on synthetic datasets with varying percentages of missing observations and image observation noise levels illustrates issue 1 above. He found that his method performed well for missing observation percentages below some threshold, typically 50% for datasets with 2.0 pixel image observation noise and 70% for datasets with 0.1 pixel image observation noise, and failed “catastrophically” for datasets with higher percentages.

Weng, *et al.*[60] investigated 2 by deriving an analytical result describing the accuracy of optimal shape and motion estimates from image measurements as the percentage of missing observations varies. They show that for a scenario including some reasonable simplifying assumptions, the error variance in both the estimated shape and motion increases as $O(n/(f^2))$, where the n is the number of images and f is the average number of images that each point was visible in. That is, the error variances increase rapidly with the percentage of missing observations.

Issue 3 has been largely ignored in the shape-from-motion community. But, the SLAM literature is largely concerned with recursively estimation of a robot’s position in the plane from range measurements and odometry, in scenarios where the robot revisits a previously mapped location. Section 2.8 and Chapter 7 below consider this problem in the context of image-based motion estimation.

As mentioned at the beginning of this section, algorithms exist that estimate motion and structure over long sequences by performing shape-from-motion on subsets of 2 or 3 images from the sequence, and concatenating these solutions to estimate the overall motion. While these methods can often generate highly accurate estimates, they are ad hoc in that they do not attempt to produce statistically optimal estimates, and they do not produce a measure of the error in the estimates.

The recent system described by Nistér, *et al.*[45], already mentioned in Section 2.6 above, is a successful example of this type that deserves special mention. This system generates robust shape-and-motion estimates from long image sequences using a tuned combination of:

- simultaneous estimation of shape and motion from pairs or triples of images in the sequence
- triangulation of scene points from previously estimated camera positions, or when stereo images are available, from stereo images
- estimation of camera positions from previously estimated scene points

This system works in real-time by employing efficient algorithms for, and highly optimized implementations of, RANSAC[43], 3-point pose relative to known points, shape and motion from 5 points in two and three images[42][44], scene point triangulation, and feature extraction and matching[45]. The fast algorithms for RANSAC and 5-point shape-from-motion, in particular, are enabling technologies that are likely to be used in many future systems.

Addressing issue 2 above, drift in the estimates, in a primary concern in both Nistér, *et al.*'s work and the work presented here, but the two systems differ in their approach to this problem. In Nistér, *et al.*'s system, drift is minimized using (1) the fast RANSAC and the 5-point shape-from-motion, which allow a real-time search for the most accurate possible matches between image pairs, (2) by heuristics for triangulating points from the longest possible baseline, and (3) heuristics for preventing the propagation of gross errors in the shape and motion estimates.

The work described here investigates approaches to this problem using omnidirectional images, which produce motion estimates less sensitive to observation errors and allow points to be tracked longer in the image sequence; the integration of image and inertial measurements, which eliminates some sensitivity to image observation noise and allows two components of the sensor rotation to be established without drift; constrained feature tracking, which reduces observation noise and gross errors; and feature reacquisition for long traverses, which limits drift when the sensors revisit previously mapped locations. The last of these is discussed in more detail in the next section.

2.8 Long-term image-based motion estimation

Existing methods for estimating motion from image measurements are not suitable for the large-scale problems that arise in autonomous vehicle navigation and modeling from video applications because, as described in the previous section, these methods drift when observations are missing. When the camera revisits previously mapped locations, however, drift should be limited by:

1. Recognizing that a previously mapped location has been revisited.
2. Using observations of revisited features to reduce drift by, for example, closing the loop in the motion estimates.

The second problem has been extensively studied in the SLAM community, whereas the first problem is unique to image-based motion estimation.

As Section 2.6 above mentions, the tracker in Chapter 6 uses SIFT keypoints and their associated feature vectors to estimate the epipolar geometry between adjacent images in an image sequence. SIFT keypoint locations can often be extracted even in the presence of image transformations such as scale, rotation, and affine transformations, and their associated feature vectors are often matched to the corresponding SIFT feature vector from a different viewpoint using a relatively simple method. So, SIFT keypoints are also a good candidate for reacquiring revisited features as well as features in adjacent images. Chapter 7 below describes a proof of concept system for estimating motion with limited drift in a bounded area, which uses SIFT keypoints for reacquisition.

Se, Lowe, and Little[53] describe a system for simultaneous localization and mapping from trinocular stereo images and odometry that operates by matching SIFT keypoints both between stereo images and between images taken at adjacent times. This system's map contains the SIFT keypoints' estimated three-dimensional positions, which are maintained using individual Kalman filters. While reacquisition and closing the loop are not demonstrated, adding some basic functionality for closing the loop to Se, *et al.*'s system would be a natural extension.

2.9 Background: conventional camera projection models and intrinsics

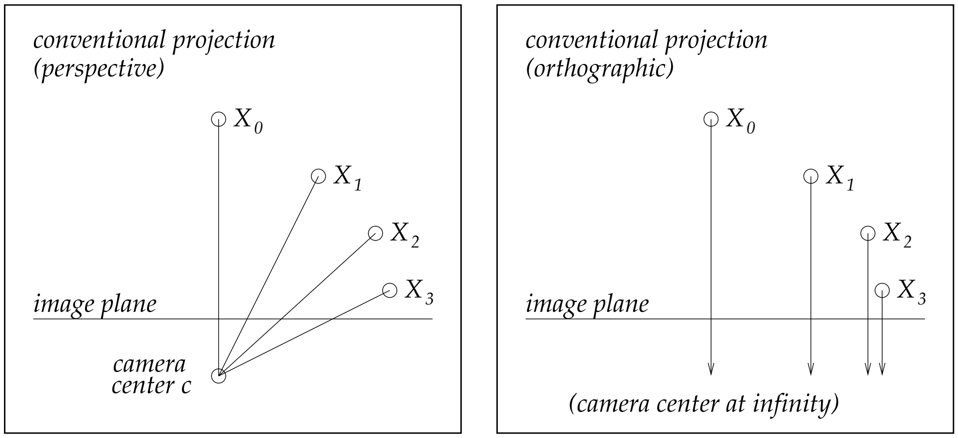
Conventional cameras are normally modeled using a two-step process. In the first step, normalized coordinate system projections are computed by intersecting the ray from the scene point to the camera center with the camera's image plane, as shown in Figure 2.1(a). If we assume that the distance from the camera center to the image plane is one unit, then this method produces normalized perspective projections. Suppose that:

- the camera coordinate system origin is at the camera center
- the camera coordinate system x, y, and z axes are aligned with the right image direction, aligned with the down image direction, and pointing into the scene, respectively

Then, the normalized perspective projection of a scene point $X = [X_x, X_y, X_z]^{\mathrm{T}}$ expressed in the camera coordinate system is:

$$\pi_{\text{perspective}}\left(\begin{bmatrix} X_x \\ X_y \\ X_z \end{bmatrix}\right) = \begin{bmatrix} X_x/X_z \\ X_y/X_z \end{bmatrix} \tag{2.1}$$

If we instead assume that the camera center is infinitely far from the image plane, then intersecting the ray from the scene point to the image center with the image plane instead produces the orthographic projection, as shown in Figure 2.1(b). The orthographic projec-



(a) (b)

Figure 2.1: Normalized perspective projections are found by intersecting the ray from the scene point to the camera center with the image plane (a). Orthographic projection, which does not model the effects of the scene points' distances from the camera, can be treated as a special case of perspective projection in which the camera center is infinitely far away (b).

tion is:

$$\pi_{\text{orthographic}}\left(\begin{bmatrix} X_x \\ X_y \\ X_z \end{bmatrix}\right) = \begin{bmatrix} X_x \\ X_y \end{bmatrix} \tag{2.2}$$

As equation (2.2) shows, orthographic projection does not model the effects of the scene points' relative distances from the camera, and is therefore normally used to model cameras viewing a scene where the range of depths is small compared to the overall scene depth.

In the second step, the camera's known intrinsics are used to remap the normalized coordinate system projections to image coordinate system projections, which are the locations of the scene points in the actual image. This remapping accounts for those camera effects that can be computed without knowledge of the points' three-dimensional locations, such as camera focal length and radial distortion.

For problems like motion estimation that require computing many projections and their derivatives, work in the normalized coordinate system is more efficient than in the image coordinate system. In this case, the intrinsics are used in reverse to map image observations to normalized coordinate system observations. Then motion estimation, for example, can be performed by minimizing residuals in the normalized coordinate system, as discussed in Section 2.10 below.

Working in the normalized coordinate system also requires that covariances associated with image system projections be normalized. If x_i is an image projection, C_i the covariance associated with x_i, and x_n the corresponding normalized projection, then a corresponding normalized covariance can found using:

$$C_n = \frac{\partial x_n}{\partial x_i} C_i \left(\frac{\partial x_n}{\partial x_i} \right)^{\mathrm{T}} \tag{2.3}$$

Throughout this work, the intrinsics model described by Heikkilä and Silvén[27] has been used to convert between normalized and image coordinate system projections, and a publicly available algorithm for estimating the parameters of this model from images of a known target has been used to perform the camera calibration. Heikkilä and Silvén's model includes separate focal lengths for the x and y image directions, which together account for both focal length and rectangular pixels; skew between the x and y image axes; a three-term polynomial radial distortion model; and a tangential distortion model. Average calibration residuals between 0.2 pixels and 1 pixel are typical with this intrinsics model and calibration algorithm.

The average residuals produced by Heikkilä and Silvén's model are a significant improvement over Tsai's[58][59] older, but widely used, model and calibration. However, the residuals for individual calibration points using Heikkilä and Silvén's model are often 2 pixels or more, suggesting that this model does not completely model the camera over the entire field of view. Because motion estimation from images can be sensitive to projection errors of much less than 2 pixels, it would be worthwhile to investigate more accurate camera intrinsics models in the future.

2.10 Background: bundle adjustment

Bundle adjustment[61][57] and nonlinear shape-from-motion[55] are similar methods for simultaneously estimating camera motion and sparse scene structure from projections in an image sequence. These are batch algorithms, meaning that they use all the observations from the image sequence at once, and can produce highly accurate estimates if the camera model and projection data are accurate.

This section briefly reviews a bundle adjustment algorithm suitable for use with any conventional or omnidirectional projection model. This algorithm is used in several subsequent sections:

- In Section 3.5, the algorithm is used with the omnidirectional projection model from Section 3.3 to evaluate the omnidirectional camera calibration approach in Section 3.4.
- The bundle adjustment error term forms one leg of the batch algorithm for estimating motion from image and inertial measurements described in Section 4.3.
- The algorithm is used on a prefix of the image sequence to initialize the recursive, image-only motion estimation algorithm described in Section 4.4.7.
- In Chapter 5, the algorithm is used to generate image-only motion estimates from both conventional and omnidirectional images for comparison with the image-and-inertial motion estimates.
- In Chapter 6, the algorithm is used to estimate epipolar geometry and detect mistracking within the tracker.

The algorithm uses Levenberg-Marquardt to minimize the error between observed projections and predicted projections, with respect to the camera and point estimates. This section describes the error function; see [48] for the details of Levenberg-Marquardt, which is widely used.

The error term is:

$$\chi^2 = \sum_{i,j} D_{i,j}(\pi(W_{\rho_i,t_i}(X_j)), x_{ij}) \tag{2.4}$$

Equation (2.4) specifies an image reprojection error given the six degree of freedom camera positions and three-dimensional point positions. In this error, the sum is over i and j, such that point j was observed in image i. x_{ij} is the observed projection of point j in image i. ρ_i and t_i are the rotation Euler angles and translation specifying the camera-to-world coordinate system transformation, respectively, at the time of image i; and W_{ρ_i,t_i} is the corresponding world-to-camera transformation. X_j is the world coordinate system location of point j, so that $W_{\rho_i,t_i}(X_j)$ is location of point j in camera coordinate system i.

π gives the image projection of a three-dimensional point specified in the camera coordinate system. For conventional images, π might be the perspective projection or orthographic projection. Section 3.3 below describes a projection model suitable for use with omnidirectional images.

The individual distance functions $D_{i,j}$ are Mahalanobis distances. If the operator π produces image coordinate system projections (i.e., if π accounts for the camera focal length and other intrinsics), then reasonable choices for the associated covariances are directional covariances determined using image texture in the vicinity of each feature[7][34], or isotropic covariances with $\sigma = 1$ pixels or $\sigma = 2$ pixels. However, π is normally chosen to give projections in the normalized coordinate system. In this case, suitable covariances can be found by converting the directional or isotropic image coordinate system covariances to normalized covariances as described in Section 2.9.

Levenberg-Marquardt is an iterative algorithm, and to ensure convergence, the initial estimate provided to the method must be sufficiently close to the solution. If a recursive motion estimation algorithm is applicable, then a good recursive solution is a suitable initial estimate for bundle adjustment. Each of the sections below that includes bundle adjustment results also describes how the initial estimates were computed.

Chapter 3

Motion from noncentral omnidirectional image sequences

3.1 Overview

Recent omnidirectional camera designs aim a conventional camera at a mirror that expands the camera's field of view, typically to $360°$ in azimuth and $90°$ to $140°$ in elevation. For camera motion estimation, there are two potential advantages to this wide field of view. First, the wide view eliminates the sensitivity in motion estimates that may occur with a conventional camera, such as the confusion between small rotational and small translational motions. Second, tracked points are likely to be visible longer in an omnidirectional image sequence, which is helpful for motion estimation from long sequences.

To fully exploit these advantages, however, the omnidirectional camera's projection (i.e., imaging) process must be accurately modeled. This chapter describes an approach for modeling omnidirectional cameras, for calibrating omnidirectional cameras using this model, and for estimating camera motion from omnidirectional images using this model. Unlike existing methods, the projection model described in this chapter accommodates both single viewpoint omnidirectional cameras and the wider space of non-single-viewpoint, or *noncentral* cameras, described below. The method also accommodates known six degree of freedom misalignment between the conventional camera and mirror that compose the omnidirectional camera.

This chapter is organized as follows. Single viewpoint and noncentral omnidirectional cameras are briefly introduced in Section 3.2, and Section 3.3 describes the general omnidirectional projection model. Sections 3.4 describes omnidirectional camera calibration using this model. Experimental results evaluating the model and calibration are given in

Section 3.5.

The algorithms for estimating camera motion from image and inertial measurements described in Chapter 4 below also accommodate the omnidirectional projection model described in this chapter. Chapter 5 uses this model to evaluate the relative advantages of using omnidirectional images and of combining conventional image and inertial measurements for motion estimation.

3.2 Omnidirectional cameras

In *single viewpoint* omnidirectional cameras, the mirror profile and the relative position between the conventional camera and mirror are chosen so that scene rays reflected into the camera's center by the mirror also intersect at a common point inside the mirror, as shown in Figure 3.1(a). In the original single viewpoint design described by Nayar[41], for example, scene rays reflecting from a parabolic mirror aligned to meet at an orthographic camera's center at infinity also intersect at the parabola's focus inside the mirror. This design is illustrated in Figure 3.1(b). The single viewpoint property allows the omnidirectional image to be unwarped to produce correct conventional (i.e., perspective) views of part of the omnidirectional camera's view. In addition, the intersection point inside the mirror can be adopted as the camera's center for defining and computing the epipolar geometry between two omnidirectional views.

Single viewpoint designs are the most widely used in the vision community for these reasons. But the single viewpoint design space, explored by Baker and Nayar[1], is small compared to that of non-single-viewpoint, or *noncentral*, omnidirectional cameras. As shown in Figure 3.1(c), noncentral cameras are those in which scene rays reflecting from the mirror into the camera center do not also intersect in a common point inside the mirror.

Two specific noncentral camera scenarios are interesting. First, the image resolution in single viewpoint images can vary greatly between the image center and the image periphery. In Nayar's design, for instance, the resolution varies by a factor of four between the center and periphery. On the other hand, dropping the single viewpoint constraint allows the design of a noncentral *equiangular* camera, where objects spaced θ degrees apart in elevation relative to the mirror's apex are always spaced d pixels apart along the image's radial lines[46]. Figure 3.2 illustrates this idea. Figures 3.3 and 3.4 show two actual equiangular cameras with example images generated by them.

Second, a single viewpoint camera design only produces single viewpoint images if the conventional camera and mirror are precisely aligned. If there is some known six degree of freedom misalignment between the conventional camera and mirror, then the camera

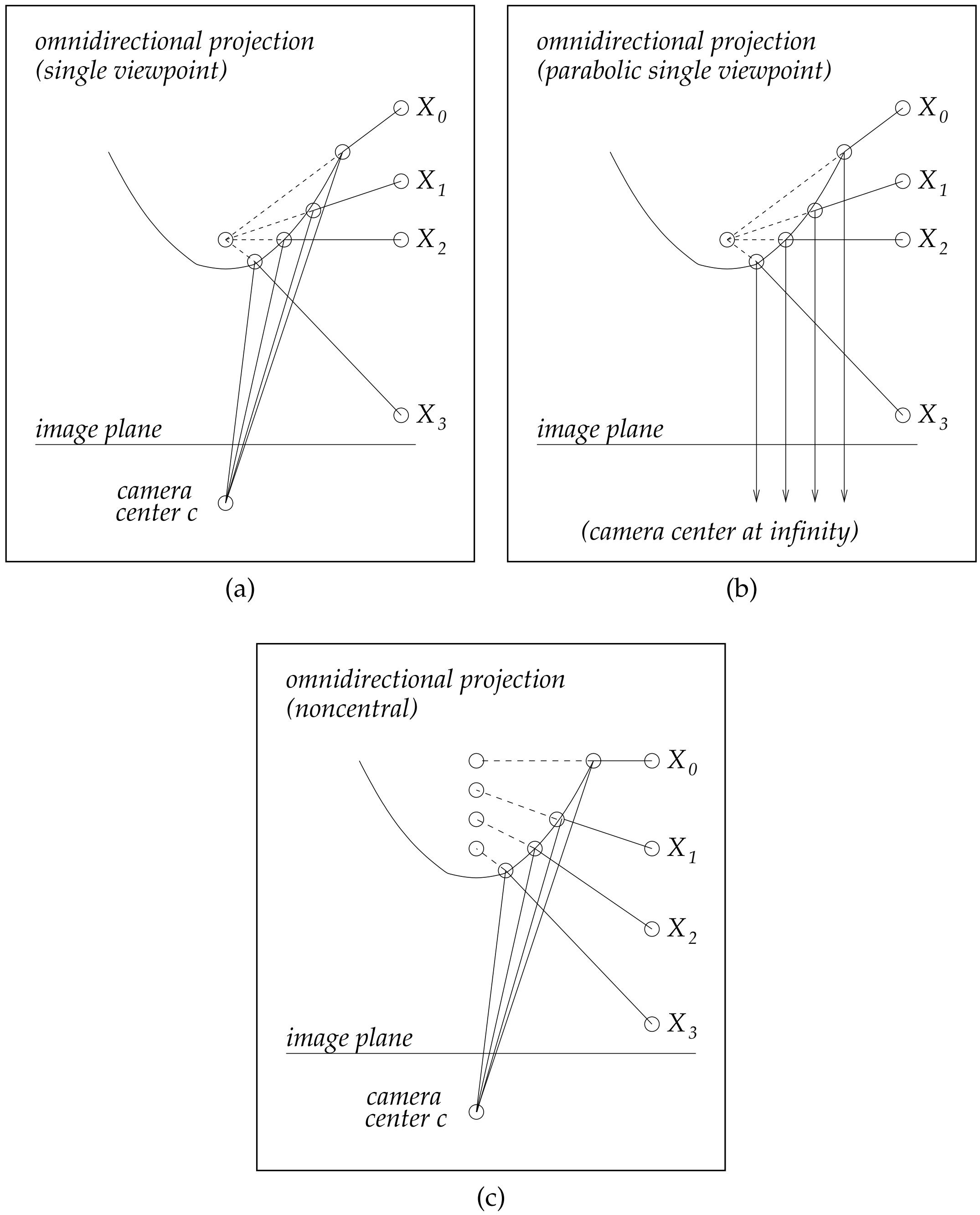
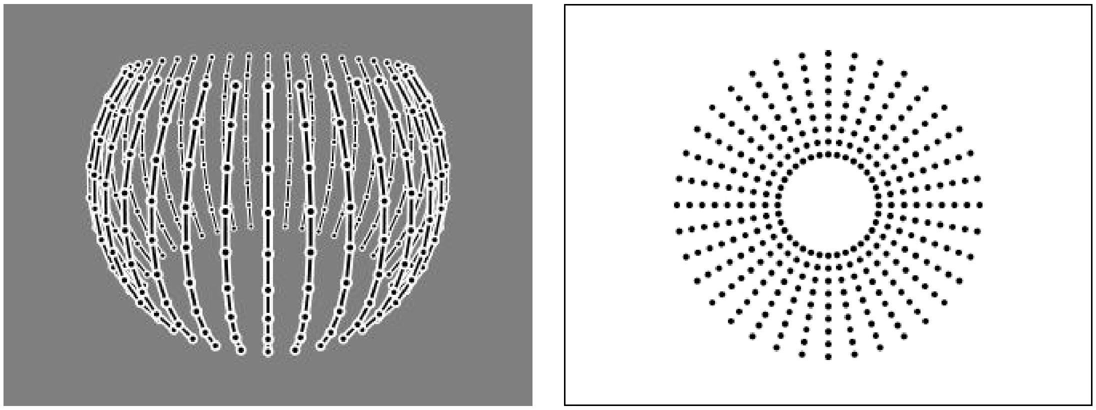
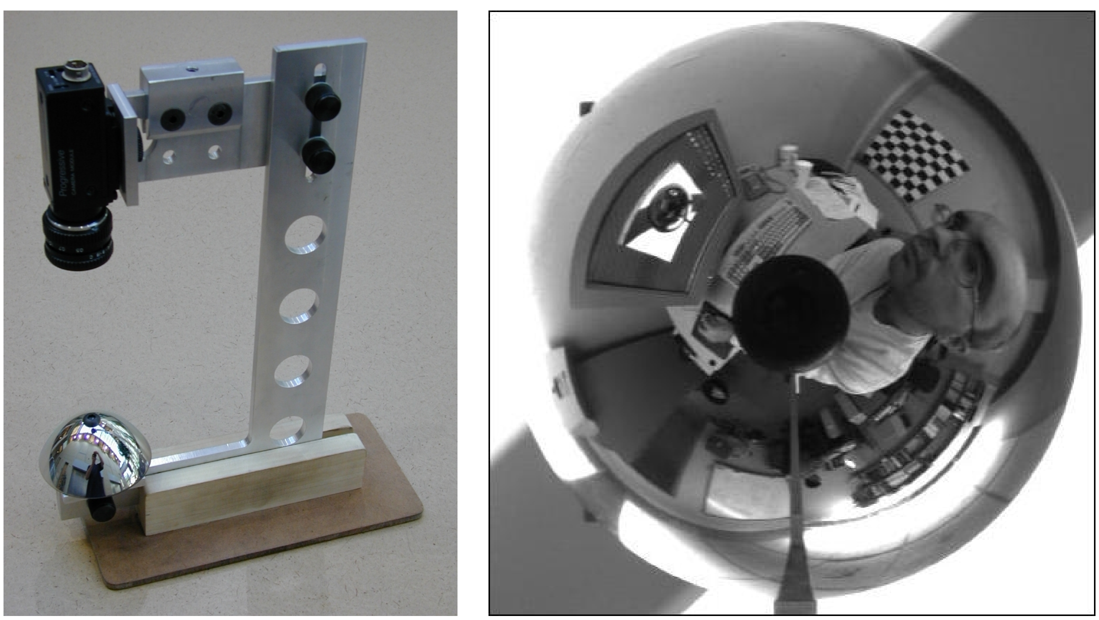


Figure 3.1: In single viewpoint omnidirectional cameras, scene rays reflected into the camera's center by the mirror also intersect at a common point inside the mirror, as shown in (a) and (b). In noncentral omnidirectional cameras, the scene rays do not meet at a common point inside in the mirror (c).



(a) (b)

Figure 3.2: A scene consisting of points on longitudinal lines is shown in (a); within each longitudinal line, the points are equally space in angle about the origin. An equiangular omnidirectional camera whose mirror is placed at the origin of the scene images the points at equally spaced positions along the radial lines, as shown in (b).



(a) (b)

Figure 3.3: An equiangular omnidirectional camera consisting of a CCD camera attached to a mirror by a rig that allows the relative rotation and translation between the mirror and camera to be adjusted (a). An omnidirectional image produced by this design (b). In (b), the camera is upright, between the monitor on the left and the person on the right.

(a) (b)

Figure 3.4: An equiangular omnidirectional camera consisting of an IEEE 1394 camera attached to a mirror by a rigid cylinder (a) and an omnidirectional image produced by this design (b). In (b), the camera is upright, between the rover on the bottom of the image and people on the top.

can be effectively modeled as a noncentral camera.

3.3 Computing omnidirectional projections

To estimate camera motion and sparse scene structure from omnidirectional images, we need to relate omnidirectional image projections to the three-dimensional positions of the corresponding scene points. From the discussion above, it is clear that a scene point X appears in the omnidirectional image at the perspective projection of the mirror point that reflects a ray from the scene point into the camera's center.

So, to find the appropriate mirror point m given X, suppose we have the following coordinate systems:

1. The conventional camera coordinate system C, defined as in Section 2.9, with the origin at the camera's center; the x axis aligned with the image right direction; the y axis aligned with the image down direction; and the positive z axis aligned with camera's forward optical axis.

2. The global mirror coordinate system M, defined with the origin at the mirror's apex; and the positive z axis aligned with the mirror's axis.

3. A local mirror coordinate system $L(m)$, defined for any point m on the mirror surface, with the origin at m; the x and y axes in the tangent plane of the mirror at m; and the z axis orthogonal to the tangent plane, facing out from the mirror.

And assume that the following are known:

1. The camera-to-world transformation that describes the three-dimensional rotation and translation of the conventional camera relative to the world coordinate system.

2. The mirror-to-camera transformation that describes the three-dimensional rotation and translation of the mirror relative to the conventional camera coordinate system. This allows us to compute the camera center in the mirror coordinate system, c_M.

3. The omnidirectional camera's mirror profile, which gives the radius r of the mirror as a function of height z above the mirror's apex. Since the mirror is a surface of revolution, this profile allows us to compute the local coordinate system $L(m)$ at m relative to the global mirror coordinate system M.

4. The three-dimensional position X of the scene point in the world coordinate system. Since the camera-to-world and mirror-to-camera transformation are also assumed in 1 and 2 above, the scene point's positions X_C and X_M in the camera and mirror coordinate systems are then also known.

Then, to find the mirror reflection point m for a particular scene point X, the constraints that m must satisfy are those describing a mirror's behavior:

1. In P is the plane containing m_M and X_M and orthogonal to the mirror tangent plane at m_M, then P must contain the reflected ray $c_M - m_M$ as well as the incoming ray $X_M - m_M$.

2. The angle of incidence that $X_M - m_M$ makes with the tangent plane must equal the angle of reflection that $c_M - m_M$ makes with the tangent plane.

These constraints are illustrated in Figure 3.5.

If the mirror and conventional camera axes are assumed to be coincident, then the azimuth θ of m_M is the azimuth of the camera coordinate system point X_C, so that finding m_M reduces to the one-dimensional problem of finding m_M's height z above the apex such that the angle of incidence equals the angle of reflection. In this one-dimensional case, z can be found using the bisection method.

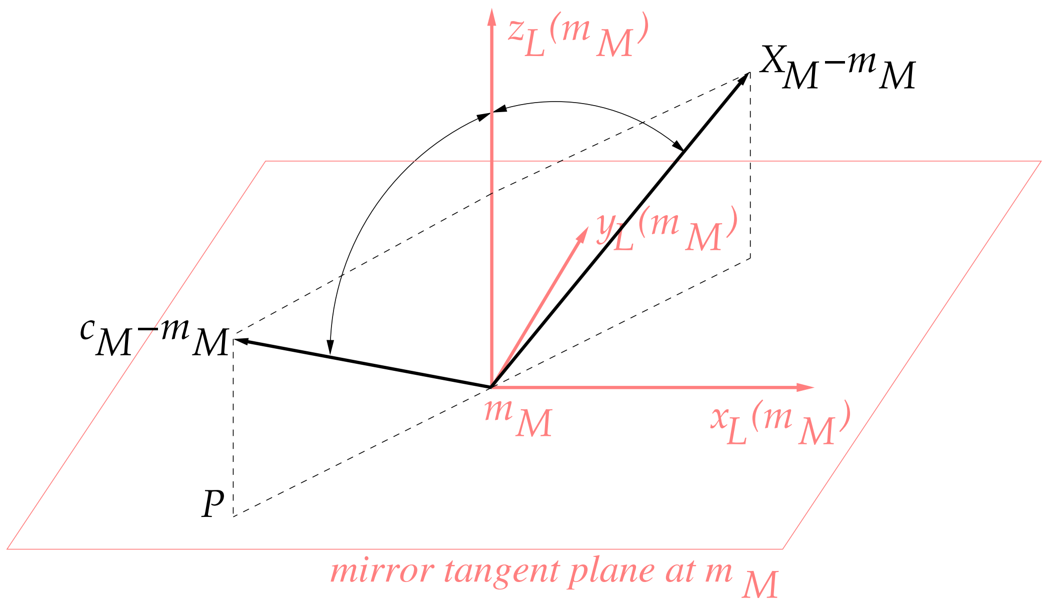


Figure 3.5: At the reflection point m for a given scene point X, the ray reflecting to c must be in the same plane P as X and m orthogonal to the tangent plane; and the angle of incidence must equal the angle of reflection.

If the mirror and conventional camera axes are not assumed to be coincident, then an azimuth θ and a height z for m_M must be found that satisfies the constraints. As we can see from examining the diagram, this is equivalent to satisfying:

$$\frac{x_L(m_M)\cdot(X_M-m_M)}{z_L(m_M)\cdot(X_M-m_M)}=\frac{-x_L(m_M)\cdot(c_M-m_M)}{z_L(m_M)\cdot(c_M-m_M)} \tag{3.1}$$

$$\frac{y_L(m_M)\cdot(X_M-m_M)}{z_L(m_M)\cdot(X_M-m_M)}=\frac{-y_L(m_M)\cdot(c_M-m_M)}{z_L(m_M)\cdot(c_M-m_M)} \tag{3.2}$$

In this two-dimensional case, θ and z can be found using Newton's method. Since Newton's method is iterative, many initial positions for the search may be required before the method converges to the mirror point m that satisfies the constraints. The current implementation uses a hierarhical method, which scans initial positions for θ and z at increasing resolution until the mirror point m is found. Searching from many initial positions is computationally expensive, and in the worst case, the search may continue indefinitely without finding the projection. However, this method makes it possible to compute correct projections even for some severe misalignments.

Figure 3.6(a) and 3.6(b) show the projections computed using this method for an equiangular camera, the scene and camera position shown in Figure 3.2(a), and two different kinds of misalignment between the camera and mirror. As mentioned above, using the hierarchical method for visiting initial positions, omnidirectional projections can be computed reliably even for some extreme misalignments between the camera and mirror, as shown in Figure 3.6(c).

The derivative of a projection with respect to X_C and with respect to the mirror-to-camera transformation are needed for motion estimation and calibration, respectively. With the method for computing projections in hand, these two derivatives can be approximated using finite differences.

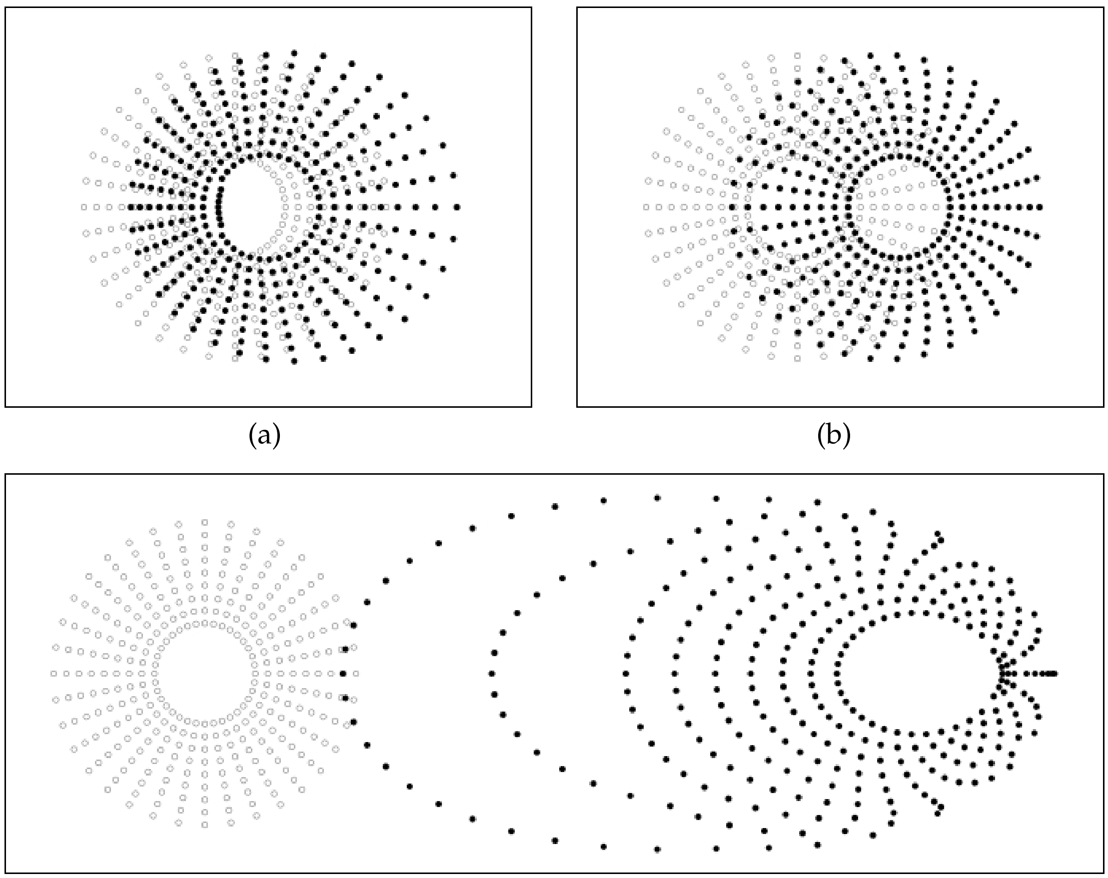
3.4 Omnidirectional camera calibration

The method for computing omnidirectional projections described in the previous section can accommodate six degree of freedom misalignment between the camera and mirror if the mirror-to-camera transformation is known. This section describes a straightforward algorithm for estimating the mirror-to-camera transformation from one image of known three-dimensional points.

Camera motion estimation benefits from an accurate model of the camera, so estimating the misalignment between the camera and mirror is worthwhile even if the misalignment is slight. But even in carefully manufactured cameras, there may be a significant misalignment. In the omnidirectional camera in Figure 3.4, for instance, there is a significant rotation between the mirror and camera because the ends of the clear cylinder holding the camera and mirror together are not cut at exact right angles to the cylinder's axis. Another example would be an incorrect distance between the mirror and conventional camera, created during assembly because the camera's center is not a physically observable location on the camera. Another more benign scenario, which is also handled by other omnidirectional camera calibration methods, is a relative x, y translation between the conventional camera's image center and the mirror axis. On the other hand, if the misalignment between the camera and mirror is thought to be negligible, the algorithm described in this section can be used to verify the alignment.

The mirror-to-camera transformation is found using an algorithm similar to the bundle adjustment algorithm described in Section 2.10 above, but employing only one image and recovering a different set of unknowns. More specifically, Levenberg-Marquardt is used to minimize:

$$\chi^2 = \sum_{i=1}^{n} \parallel x_i - \pi(W_{\rho,t}(X_i)) \parallel^2 \tag{3.3}$$



(a) (b)

(c)

Figure 3.6: The projections that result from imaging the scene in Figure 3.2 with an equiangular camera when the mirror is (a) rotated 0.2 radians about its y axis, (b) 2 cm along it x axis, and (c) 14 cm along its x axis are shown in black. In each case, the projections that result when the camera and mirror are exactly positioned are shown as a reference in gray.

where:

- π is the omnidirectional projection model described in the previous section
- n is the number of known points visible in the calibration image
- X_i is the known three-dimensional location of point i in the world coordinate system
- ρ and t are the Euler angles and translation specifying the camera-to-world transformation at the time of the calibration image
- $W_{\rho,t}$ is the corresponding world-to-camera transformation
- x_i is the observed normalized projection for point X_i

Since π is a function of the mirror-to-camera transformation, (3.3) is minimized with respect to the camera-to-world and mirror-to-camera Euler angles and translations. Since the mirror is a surface of revolution, the mirror-to-camera rotation about the mirror axis cannot be recovered and is excluded from the optimization. So, (3.3) is minimized with respect to eleven parameters total. After the calibration, the recovered camera-to-world parameters can be discarded.

Section 3.5 below evaluates the accuracy of this calibration and its benefits for different levels of misalignment between the camera and mirror.

3.5 Experimental results

3.5.1 Overview

To determine whether the calibration algorithm produces the correct mirror-to-camera transformation, and whether the full projection model improves the accuracy of motion estimation, three image sequences have been captured with increasing amounts of misalignment between the mirror and camera axes. In Sequence 1, the mirror and camera axes are aligned with the same accuracy as, or better accuracy than, we might expect from an off-the-shelf omnidirectional camera. In Sequence 2, the camera is significantly rotated about its center of projection. This is both a translation and a rotation of the mirror in the camera coordinate system, so the resulting distortion is a combination of those shown in Figures 3.6(a) and 3.6(b). Sequence 3 is similar to Sequence 2, but with a more gross rotation about the camera's center.

For each sequence, a calibration image of the known targets shown in Figure 3.7(a) has been captured. The calibration images for Sequences 1 and 3 are those shown in Figure

	Calibration A	Calibration B	Calibration C
Sequence 1	1.92	1.91	1.34
Sequence 2	4.01	3.85	1.37
Sequence 3	6.50	6.30	1.34

Table 3.1: The average reprojection errors for each sequence and calibration in the calibration experiments.

3.7(b) and 3.7(c). The image locations of the calibration targets for the calibration image in 3.7(b) are shown in 3.7(d).

For each of these sequences, three calibrations have been performed. Calibration A assumes that the mirror and camera have precisely the correct relative positions; for the mirror used in these tests, the mirror apex should be 14.0 cm from the camera center. Calibration B assumes that the mirror and camera axes are identical, but that the mirror may be vertically translated with respect to the camera away from the ideal 14.0 cm location. Calibration C makes no assumptions about the relative positions of the camera and mirror, i.e., all five degrees of freedom are recovered.

3.5.2 Correctness of the estimated parameters

Some basic statistics from these nine calibrations are shown in Tables 3.1, 3.2, and 3.3. Table 3.1 shows the average reprojection errors, i.e., the average distance in pixels between the observed target image locations and those predicted by the estimated camera-to-world and mirror-to-camera transformations. As one might expect, these show calibration models A and B, which assume that the mirror and camera axes are identical, model the observations more poorly as these two axes are rotated relative to each other, whereas model C, i.e., the full calibration, is a sufficiently strong model in these cases.

Tables 3.2 and 3.3 gives the resulting estimates of the mirror-to-camera transformation, which differ primarily in the x translation, and the estimated standard deviations. The standard deviations were computed assuming a 0.25 pixel standard error in the target location observations. Although one might guess that simultaneously estimating the camera-to-world and mirror-to-camera transformations from a single image would produce estimates that are sensitive to observation errors, these standard deviations, which are insignificant compared to the range of possible parameter values, show that this is not the case.

One independent check on the physical correctness of the full calibration can be performed by comparing the observed image location of the mirror apex to the reprojection of

	rotation about x (radians)	rotation about y (radians)
Sequence 1	0.0455 ± 0.0023	0.0065 ± 0.0022
Sequence 2	0.0455 ± 0.0023	-0.0086 ± 0.0022
Sequence 3	0.0439 ± 0.0023	-0.0142 ± 0.0022

Table 3.2: Estimates and standard deviations for the rotation parameters of the full calibration.

	t_x (cm)	t_y (cm)	t_z (cm)
Sequence 1	$-0.02 \pm 7.1 \times 10^{-3}$	$0.07 \pm 7.3 \times 10^{-3}$	$14.1 \pm 2.8 \times 10^{-2}$
Sequence 2	$-0.23 \pm 7.0 \times 10^{-3}$	$0.05 \pm 7.1 \times 10^{-3}$	$14.1 \pm 2.8 \times 10^{-2}$
Sequence 3	$-0.38 \pm 7.0 \times 10^{-3}$	$0.06 \pm 7.1 \times 10^{-3}$	$14.1 \pm 2.8 \times 10^{-2}$

Table 3.3: Estimates and standard deviations for the translation parameters of the full calibration.

the apex predicted by the mirror-to-camera transformation. These reprojections are shown in Table 3.4, along with the observed locations. The observed location has been taken to be the image location of the center of the mirror screw, which attaches the mirror to the rig and passes through the mirror axis. In each case the predicted and observed centers are quite close.

3.5.3 Effect of calibration on estimated motion and structure

To help determine whether the mirror-to-camera transformation recovered by the calibration is physically accurate, and whether modeling the mirror and camera misalignment is worthwhile for different levels of misalignment, the bundle adjustment accuracy for each combination of sequence and calibration model has been computed. Each of the three sequences consists forty-one images, captured by moving the camera by hand in a "U" shape on the calibration lab's optical bench. Therefore, the images are similar to the calibration images shown in Figure 3.7, but show the scene from a large range of views. The "U" motion was chosen to ensure some camera motion parallel to each of the walls and therefore improve the estimation accuracy for points on all walls. Manually corrected Lucas-Kanade was used to track the positions of the calibration targets through the image sequence.

The results are summarized in Table 3.5. For each of the nine tests, the average reprojection error is shown before the slash, and an average range error is shown for each test after the slash. The range errors are the average error in distance to each of the calibration targets, as measured from the middle camera position in the forty-one-image sequence. The

	Predicted	Observed	Error Distance
Sequence 1	(317.0, 250.4)	(315.5, 245.0)	4.8 pixels
Sequence 2	(283.2, 248.4)	(280.5, 244.0)	5.2 pixels
Sequence 3	(258.5, 248.9)	(258.5, 244.0)	4.9 pixels

Table 3.4: The observed and estimated locations of the mirror apex in each of the full calibrations.

	Calibration A	Calibration B	Calibration C
Sequence 1	0.40 pixels / 3.3 cm	0.41 pixels / 3.5 cm	0.37 pixels / 2.1 cm
Sequence 2	1.1 pixels / 9.8 cm	1.1 pixels / 9.5 cm	0.42 pixels / 1.6 cm
Sequence 3	1.9 pixels / 15.8 cm	1.92 pixels / 15.4 cm	0.37 pixels / 1.6 cm

Table 3.5: Average reprojection error and point range error for each sequence.

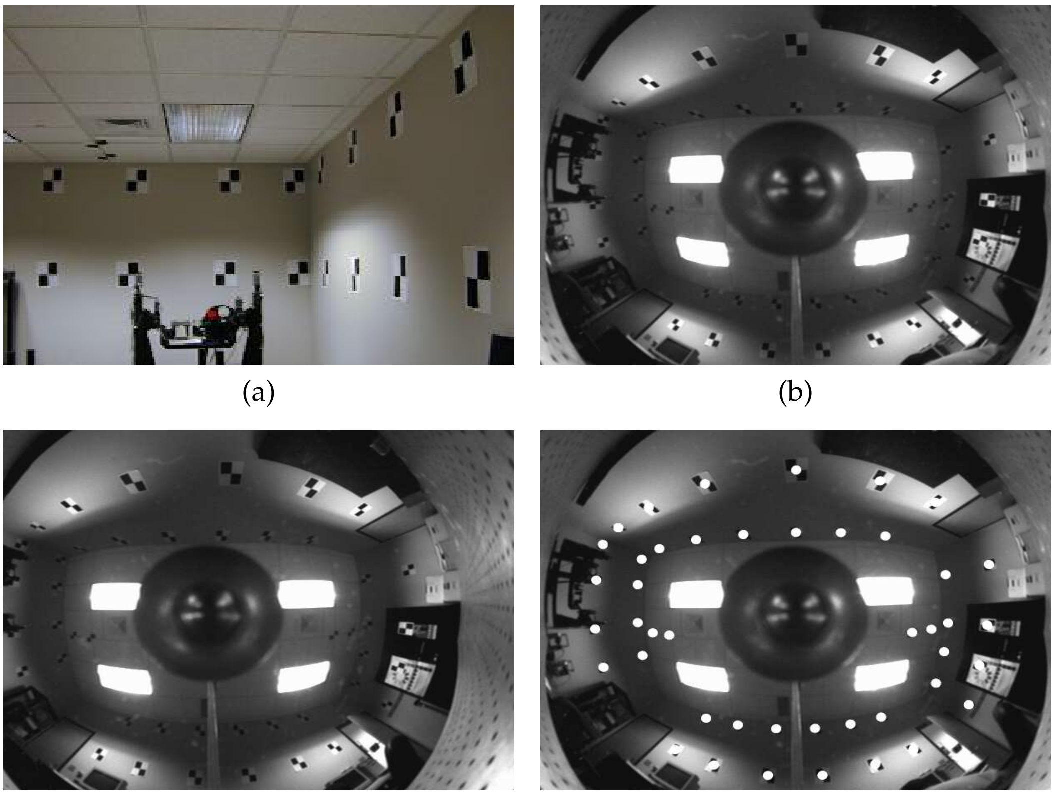
results closely follow the pattern of Table 3.1: range is recovered accurately in Sequence 1 for all three calibration models, and in all sequences for the full calibration model.

3.6 Discussion

This chapter has presented a new projection model for catadioptric omnidirectional camera that accommodates noncentral cameras and misalignment between the camera and mirror, and a calibration that determines the misalignment from a single image of known three-dimensional points. Together, the projection model and calibration can be used to extend existing shape-from-motion algorithms to omnidirectional images while achieving high accuracy.

The experiments indicate that one image is sufficient to perform the calibration, that the calibration is beneficial even in the case where the camera and mirror alignment is accurate, and that the calibration allows accurate shape and motion to be estimated even when the misalignment between the camera and mirror is severe.

In Chapter 5 below, the relative advantages of motion from omnidirectional images, and motion from image and inertial measurements, are investigated. The results indicate that these two modalities are roughly matched in their ability to increase the accuracy of motion estimation. However, omnidirectional images create some difficulties that motion from image and inertial measurements does not. For example, tracking in omnidirectional images can be difficult, whereas a robust tracker for conventional images is presented in Chapter 6. These issues are explored in more detail in the conclusion in Chapter 8.



(a) (b)

(c) (d)

Figure 3.7: The known calibration points for the experiments are shown in (a), along with the calibration images for two of the camera test configurations in (b) and (c). In (b), the mirror and camera are closely aligned. In (c), the mirror and camera are severely misaligned; this is most easily seen by comparing the left and right edges of the images in (b) and (c). Image (d) shows the observed target locations for the image in (b), which were specified by hand.

Chapter 4

Motion from image and inertial measurements: algorithms

4.1 Overview

Cameras and inertial sensors are each good candidates for autonomous vehicle navigation, modeling from video, and other applications that require six degree of freedom motion estimation. In addition, cameras and inertial sensors are good candidates to be deployed together, since in addition to the obvious advantage of redundant measurements, each can be used to resolve the ambiguities in the estimated motion that result from using the other modality alone. For instance, image measurements can counteract the error that accumulates when integrating inertial readings, whereas inertial sensors can eliminate the errors in motion that result from estimating motion from too few image measurements.

This chapter considers the specific problem of estimating sensor motion and other unknowns from image, gyro, and accelerometer measurements, in environments without known fiducials and without external positioning systems such as GPS. Accurate motion estimation under these circumstances has a number of potential applications in autonomous navigation underwater, indoors, in rubble, in the urban canyon, or on Mars, which all preclude the use of global positioning; and in modeling from video. Some important potential applications preclude the use of precise inertial navigation systems, such as micro air vehicle navigation, where weight and cost are limiting factors. So, this work specifically focuses on the use of lightweight and inexpensive inertial sensors.

This chapter presents two algorithms for estimating sensor motion and scene structure from image and inertial measurements. The first is a batch algorithm that generates estimates of the sensor motion, scene structure, and other parameters by considering all of the image and inertial measurements simultaneously. In many applications, this batch estimate is of interest in its own right. In others, the batch estimate is important in understanding the best quality we can expect given a particular sensor configuration, vehicle motion, environment, and set of observations, and in measuring the inherent sensitivity of the estimate with respect to random observation errors.

Because the batch method uses all of the measurements from an observation sequence simultaneously, it requires that all of the observations be available before computation begins. The second algorithm is a recursive method that estimates sensor motion, scene structure, and other parameters from image, gyro, and accelerometer measurements as they become available, and is therefore suitable for long or "infinite" image sequences. This algorithm is a multirate method, meaning that image measurements and inertial measurements are processed by separate update steps, which allows the higher rate of inertial measurements to be exploited. Unlike many methods for motion estimation that use tracked image point features as measurements, the recursive method also includes a mechanism for incorporating points that become visible after initialization while maintaining an accurate state covariance estimate. This capability is essential for operation on most real image sequences.

This chapter is organized as follows. Section 4.2 reviews the complementary nature of image and inertial measurements in more detail. Sections 4.3 and 4.4 detail the batch and recursive algorithms, respectively. An experimental evaluation of these algorithms is given in Chapter 5 below.

4.2 Motion from image or inertial measurements alone

Six degree of freedom motion estimates can be generated from either image or inertial measurements. Integrating gyros and accelerometer outputs once and twice, respectively, produces estimates of the sensors' rotation and translation. Similarly, camera motion and sparse scene structure can be simultaneously estimated from image measurements using a method such as the bundle adjustment algorithm described in Section 2.10 above.

However, both approaches are error prone. Both unmodeled nonlinearity and noise in inertial sensor outputs cause these estimates to drift with time. In addition, gyro and accelerometer outputs depend on biases that change with time, and accelerometer outputs depend on the orientation of the sensor relative to gravity as well as the acceleration. So, these sensors cannot produce accurate motion estimates without some means of correction and of estimating these additional unknowns.

On the other hand, simultaneous estimation of camera motion and sparse scene structure from images is brittle in practice. For many motions or environments, the estimates from these algorithms can be sensitive to random observation errors and errors in calibration. In image sequences where features enter and leave the sequence at a high rate, estimates from shape-from-motion can be sensitive to the number of tracked features and the rate of feature reextraction. Furthermore, motion estimates in this scenario will drift, just as estimates from inertial sensors will drift, because no one feature serves as a global reference of the position. In addition, image observations alone provide no mechanism for determining the global scale of the estimate.

In both cases, the estimated motion can have gross errors. So naive methods for combining image and inertial measurements, such as averaging the estimates produced by each modality individually, are only likely to produce a third poor estimate. Instead, algorithms for estimating motion from image and inertial measurements should only generate estimates that are consistent with both sets of observations simultaneously. The algorithms in this chapter take this approach.

4.3 Batch algorithm

4.3.1 Overview

The batch algorithm for motion estimation from image and inertial measurements finds estimates of the sensor motion and other unknowns using the entire observation sequence from a camera, rate gyro, and accelerometer simultaneously. As shown in Figure 4.1(a), the algorithm estimates the sensor rig rotation, translation, and linear velocity at the time of each image; the three-dimensional position of each point observed in the image sequence; the gravity vector with respect to the world coordinate system; and the gyro and accelerometer biases.

More specifically, the algorithm uses Levenberg-Marquardt to minimize a combined image and inertial error function. Since Levenberg-Marquardt is widely used, this section concentrates on the error function; see [48] for a discussion of Levenberg-Marquardt. The error function is:

$$E_{\text{combined}} = E_{\text{image}} + E_{\text{inertial}} + E_{\text{prior}} \tag{4.1}$$

The image error E_{image} is the χ^2 bundle adjustment error described in Section 2.10. The E_{inertial} and E_{prior} terms of this error are described in the following subsections.

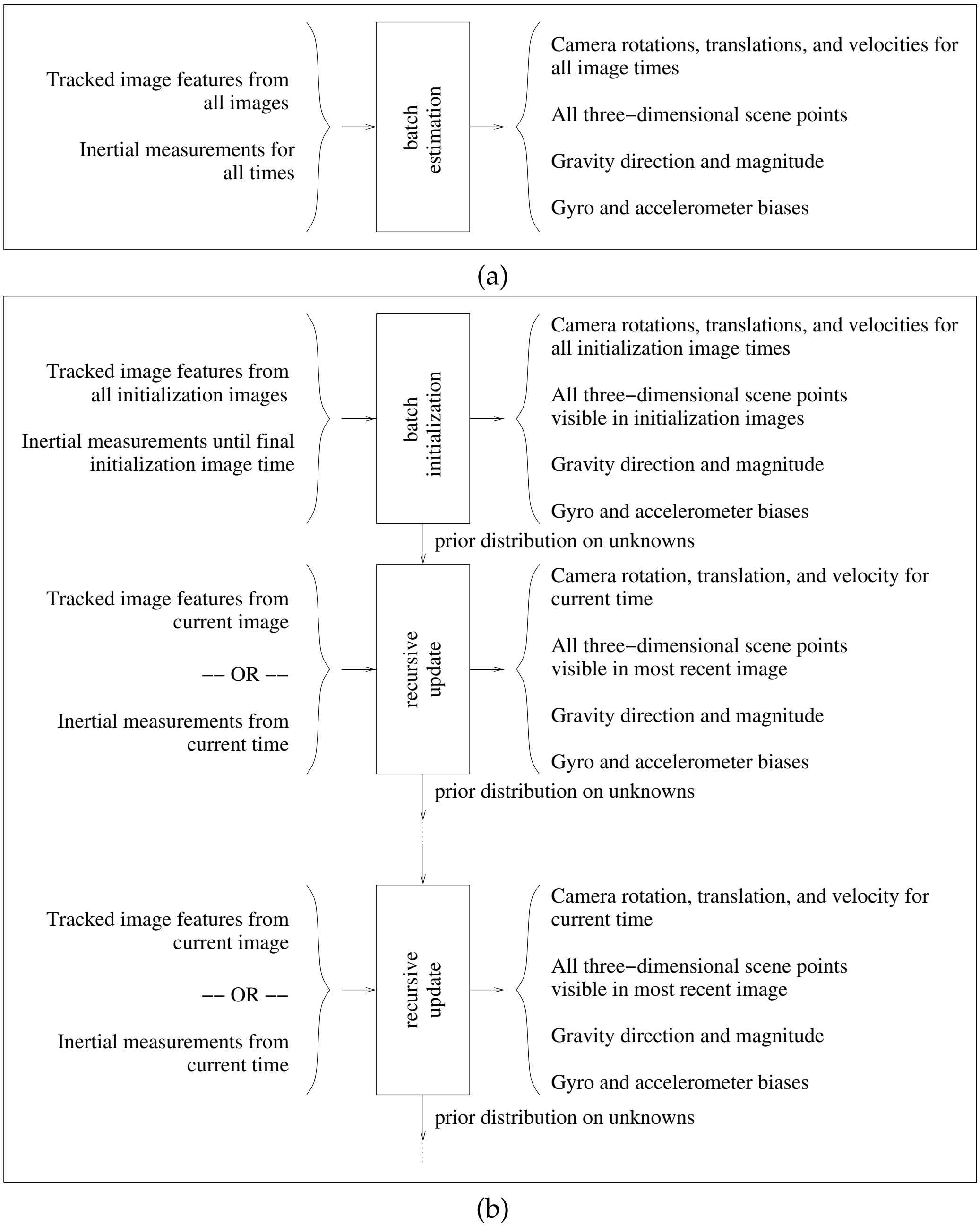


Figure 4.1: The batch algorithm for estimating motion and other unknowns from image and inertial measurements, shown in (a), uses the entire observation history at once. The recursive algorithm, shown in (b), uses the batch algorithm on a short prefix of the observations to initialize the estimates and their prior distribution, and recursively updates the estimates when subsequent observations arrive.

4.3.2 Inertial error

The inertial error term is:

$$
\begin{aligned}
E_{\text{inertial}} &= \sum_{i=1}^{f-1} D\left(\rho_i, I_\rho(\tau_{i-1}, \tau_i, \rho_{i-1}, b_\omega)\right) \\
&+ \sum_{i=1}^{f-1} D\left(v_i, I_v(\tau_{i-1}, \tau_i, \rho_{i-1}, b_\omega, v_{i-1}, g, b_a)\right) \\
&+ \sum_{i=1}^{f-1} D\left(t_i, I_t(\tau_{i-1}, \tau_i, \rho_{i-1}, b_\omega, v_{i-1}, g, b_a, t_{i-1})\right)
\end{aligned} \tag{4.2}
$$

E_{inertial} gives an error between the estimated positions and velocities and the incremental positions and velocities predicted by the inertial data. Here, f is the number of images, and τ_i is the time image i was captured. ρ_i and t_i are the camera rotation and translation at time τ_i, just as in the equation for E_{image} above. v_i gives the camera's linear velocity at time τ_i. g, b_ω, and b_a are the world coordinate system gravity vector, gyro bias, and accelerometer bias, respectively.

I_ρ, I_v, and I_t integrate the inertial observations to produce estimates of ρ_i, v_i, and t_i from initial values ρ_{i-1}, v_{i-1}, and t_{i-1}, respectively. The camera coordinate system angular velocity and world coordinate system acceleration are assumed remain constant over the intervals between consecutive measurements, whether those measurements are image or inertial measurements. Over such an interval $[\tau, \tau']$, I_ρ is defined as follows:

$$
I_\rho(\tau, \tau', \rho, b_\omega) = r(\Theta(\rho) \cdot \Delta\Theta(\tau' - \tau, b_\omega)) \tag{4.3}
$$

where $r(\Theta)$ gives the Euler angles corresponding to the rotation matrix Θ, $\Theta(\rho)$ gives the rotation matrix corresponding to the Euler angles ρ, and $\Delta\Theta(\Delta t)$ gives an incremental rotation matrix:

$$
\Delta\Theta(\Delta t, b_\omega) = \exp\left(\Delta t \,\mathrm{skew}(\omega)\right) \tag{4.4}
$$

Here,

$$
\mathrm{skew}(\omega) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{4.5}
$$

where $\omega = (\omega_x, \omega_y, \omega_z)$ is the camera coordinate system angular velocity

$$
\omega = \omega' + b_\omega \tag{4.6}
$$

and ω' is the biased camera coordinate system angular velocity given by the gyro. Over $[\tau, \tau']$ I_v and I_t are given by the familiar equations:

$$I_v(\tau, \tau', \ldots, b_a) = v + a(\tau' - \tau) \tag{4.7}$$

and

$$I_t(\tau, \tau', \ldots, t) = t + v(\tau' - \tau) + \frac{1}{2}a(\tau' - \tau)^2 \tag{4.8}$$

where a is the world coordinate system acceleration

$$a = \Theta(\rho) \cdot (a' + b_a) + g \tag{4.9}$$

and a' is the biased, camera coordinate system apparent acceleration given by the accelerometer.

Because inertial measurements are received at a higher rate than image measurements, the intervals $[\tau_{i-1}, \tau_i]$ between images span several inertial readings. To integrate the rotation, velocity, and translation over these larger intervals where the angular velocity and linear acceleration are not assumed constant, I_ρ, I_v, and I_t divide $[\tau_{i-1}, \tau_i]$ into the subintervals $[\tau, \tau']$ demarcated by the measurement times and sequentially apply (4.3), (4.7), and (4.8) over each subinterval.

The distances D in E_{inertial} are Mahalanobis distances chosen to specify the relative importance of the image error terms in E_{image} and the rotation, linear velocity, and linear acceleration error terms in E_{inertial}. The experiments in Section 5.3 investigate the sensitivity of the batch algorithm to the choice of variances that define these distances.

Two additional comments about E_{inertial} are in order. First, note that E_{inertial} does not make any assumptions restricting the relative timing between image and inertial readings. In particular, image and inertial readings need not arrive at the same time, and can arrive at different rates, as long as each measurement is accurately timestamped. Second, a possible alternative formulation for E_{inertial} is to use discrete differences to approximate the first and second derivatives of the estimated motion, and then require these derivatives to be consistent with the observed inertial measurements. But, this formulation requires that the durations between image times be small relative to the rate at which the derivatives change. Our formulation makes no such assumption, so our error function is suitable for cases where the duration between images is long.

4.3.3 Accelerometer bias prior error

The gyro and accelerometer biases b_ω and b_a represent the angular rates and accelerations reported by the inertial sensors that correspond to zero angular velocity and zero acceleration. These values are not always zero, and are estimated as part of the minimization because the voltages produced by the sensors for zero angular rate and zero acceleration can differ with temperature and across sensor powerups.

Given observation sequences where the sensors undergo little change in orientation, gravity and accelerometer bias cannot be reliably distinguished from the observations, as equation (4.9) shows. Therefore, an accelerometer bias term is included in the combined error, which reflects the expectation that the accelerometer voltage corresponding to zero acceleration is close to the precalibrated value. The bias prior term E_{prior} is:

$$E_{\text{prior}} = f \cdot b_a^{\mathrm{T}} C_b^{-1} b_a \tag{4.10}$$

As above, f is the number of images and b_a is the accelerometer bias. C_b is the accelerometer bias prior covariance.

In the experiments, the accelerometer bias prior covariance is taken to be isotropic with standard deviations of 0.5 $\mathrm{m/s^2}$. This is roughly 1% of the 4 g range of the accelerometer used in the experiments.

4.3.4 Minimization

As described in subsection 4.3.1, the combined error function is minimized with respect to the six degree of freedom camera position ρ_i, t_i at the time of each image; the camera linear velocity v_i at the time of each image; the three-dimensional positions of the tracked points X_j; the gravity vector with respect to the world coordinate system g; and the gyro and accelerometer biases b_ω and b_a.

Since the batch algorithm uses iterative minimization, an initial estimate is required, but the algorithm converges from a wide variety of initial estimates. For many sequences, an initial estimate that works well and requires no *a priori* knowledge of the motion or other unknowns is:

- All camera positions (rotations and translations) are coincident with the world coordinate system.

- The point positions are initialized by backprojecting the image position at which they first appear from the origin to a fixed distance. in space

- The velocities, gravity, and biases are initialized to zero.

Subsequent sections refer to this initial estimate as the "blind" initial estimate, and show that in some cases the algorithm can converge from the blind estimate even if the true motion is quite different from the blind estimate. For longer sequences, the recursive algorithm described in the next section can be used to generate an initial estimate close to the final estimate, making convergence more likely or reducing the number of iterations required for convergence.

As shown in Figure 4.1 and described further in Section 4.4.2, the state estimate distribution in the recursive algorithm is initialized by applying the batch image-and-inertial algorithm to a prefix of the observation sequence. But, if a recursive estimate is recommended as the initial estimate for the batch algorithm, how can the batch algorithm be used to initialize the recursive method? When the batch algorithm is used to initialize the recursive algorithm, the blind estimate for the prefix is used as the initial estimate for the batch initialization. For the short observation sequences used in the batch initialization, the prefix motion is typically benign in that the camera positions throughout the prefix are close to the initial camera positions, and most of the points visible in the first image are visible throughout the prefix. In all of the experiments that have been performed with this algorithm, including the experiments described in this paper, the batch algorithm has converged reliably from the blind estimate for prefixes including 20, 30, 40, 50, or 60 images. Of course, if the motion during the prefix is particularly erratic, the batch method may have difficulties converging and some different initialization should be substituted.

4.3.5 Discussion

Both the batch algorithm described in this section and the recursive algorithm described in the Section 4.4 represent the three-dimensional camera orientation at the time of each image using Z-Y-X Euler angles[10]. This work has adopted this representation because every choice of Euler angles represents a valid orientation, and therefore the Euler angles resulting from an unconstrained minimization step or Kalman measurement update always represent valid orientations. It is well known, however, that each orientation including a 90 degree rotation about Y is not represented by a unique choice of Z-Y-X Euler angles, but by a range of Z-Y-X Euler angles. This lack of a unique representation can lead to numerical problems in problem instances that include such camera orientations.

For these problem instances, quaternions would be a better choice for representing orientations. Since only unit quaternions correspond to valid orientations, a normalization or some other modification is required to enforce the unit constraint after an unconstrained minimization step or Kalman measurement update. However, unlike Euler angle representations, unit quaternions do not introduce numerical problems for special orientations. A few relevant, representative methods from the literature that use quaternions for representing orientation include Szeliski and Kang's batch algorithm for estimating shape and motion from image measurements[55] and Huster and Rock's method for estimating motion relative to a single point from image and inertial measurements[30][31]. A concise discussion on the relative merits of Euler angles and quaternions for motion estimation from images is given by McLauchlan[38], who also considers a local representation that pairs a reference rotation with a small incremental rotation expressed using a minimal representation.

4.4 Recursive estimation

4.4.1 Overview

The recursive method, described in this section, is an iterated extended Kalman filter (IEKF) in which the image and inertial measurements are incorporated as soon as they arrive, in separate measurement update steps. This approach exploits the higher acquisition rate of inertial data to provide state estimate updates at the higher rate, and is more principled than possible alternatives, which include queuing the inertial data until the next image measurements are available, or assuming that image and inertial measurements are taken at the same time. The application of the IEKF to this problem is described below in subsection 4.4.2, which gives the state vector; subsection 4.4.3, which describes the state estimate time propagation; and subsection 4.4.4, which gives the image and inertial measurement update steps.

Efficiency requires that the recursive method maintain estimates of, and joint error covariances between, the three-dimensional positions of only those points visible in the current image. This limitation requires a policy for removing tracked points that have been lost from the estimated state distribution, and for adding tracked points that have been newly acquired to the estimated state distribution. Our policies for incorporating new points and for removing lost points are described in subsections 4.4.5 and 4.4.6, respectively.

4.4.2 State vector and initialization

The state vector is:

$$x(\tau) = \begin{bmatrix} \rho(\tau) \\ t(\tau) \\ X_{i,0} \\ \vdots \\ X_{i,p-1} \\ v(\tau) \\ b_\omega \\ g \\ b_a \\ \omega(\tau) \\ a(\tau) \end{bmatrix} \tag{4.11}$$

Most of the components of this state vector are the same as those described for the batch method in Section 4.3. $\rho(\tau)$ and $t(\tau)$ are the Euler angles and translation specifying the camera-to-world transformation at time τ; $X_{i,0}, \ldots, X_{i,p-1}$ are the three-dimensional locations of the tracked points visible in the most recent image; $v(\tau)$ is the linear velocity at time τ expressed in the world coordinate system; and g, b_ω, b_a are the gravity vector with respect to the world coordinate system, the gyro bias, and the accelerometer bias, respectively. The two components of the state not estimated by the batch algorithm, $\omega(\tau)$ and $a(\tau)$, are the camera coordinate system angular velocity and world coordinate system linear acceleration at time τ.

Those components of the state estimate distribution that can be estimated using the batch algorithm are initialized by applying the batch algorithm to a prefix of the observation sequence. The angular velocity and linear acceleration, which are not estimated by the batch algorithm, are initialized using the first gyro and accelerometer readings that follow the observation prefix used in the batch initialization.

4.4.3 State propagation

The algorithm assumes that the state $x(\tau)$ propagates according to:

$$\dot{x}(\tau) = f(x(\tau)) + w(\tau) \tag{4.12}$$

where w is a zero mean Gaussian noise vector with covariance Q. The nonzero components of f are $dt/d\tau = v$, $dv/d\tau = a$, and

$$\frac{d\rho}{d\tau} = \frac{d\rho}{d\Theta(\rho)} \frac{d\Theta(\rho)}{d\tau} \tag{4.13}$$

As in Section 4.3, $\Theta(\rho)$ is the camera-to-world rotation matrix specified by ρ. $d\rho/d\Theta(\rho)$ is a 3×9 matrix that can be computed from the definition of $\Theta(\rho)$, and $d\Theta(\rho)/d\tau$ is a 9×1, flattened version of

$$\Theta(\rho)\ \mathrm{skew}(\omega) \tag{4.14}$$

where $\mathrm{skew}(\omega)$ is given by equation (4.5). The noise covariance matrix Q is zero except for the 3×3 submatrices corresponding to ω and a, which are assumed to be isotropic.

Assuming that the true state propagates according to (4.12), a state estimate mean $\hat{x}(\tau)$ can be propagated using

$$\dot{\hat{x}}(\tau) = f(\hat{x}(\tau)) \tag{4.15}$$

and a state estimate covariance $P(\tau)$ propagated using

$$\dot{P}(\tau) = F(\hat{x}(\tau))P(\tau) + P(\tau)F^{\mathrm{T}}(\hat{x}(\tau)) + Q \tag{4.16}$$

where P is the error covariance estimate, F is the derivative of $f(\hat{x}(\tau))$ with respect to the state estimate $\hat{x}$, and Q is the noise covariance matrix. The nonzero blocks of F are $\partial^2\rho/\partial\tau\,\partial\rho$, which are computed numerically, and

$$\frac{\partial^2 t}{\partial\tau\,\partial v} = I_3 \tag{4.17}$$

$$\frac{\partial^2 v}{\partial\tau\,\partial a} = I_3 \tag{4.18}$$

$$\frac{\partial^2 \rho}{\partial\tau\,\partial\omega} = \frac{d\rho}{d\Theta(\rho)} \frac{d\mathrm{skew}(\omega)}{d\omega} \tag{4.19}$$

Here, $d\rho/d\Theta(\rho)$ and $d\mathrm{skew}(\omega)/d\omega$ are flattened, 3×9 and 9×3 versions of the derivatives.

4.4.4 Measurement updates

When image or inertial measurements are received, the state estimate mean and covariance are propagated from the previous measurement update time using (4.15) and (4.16), and then updated using an IEKF measurement update. For brevity, this section concentrates on the image and inertial measurement equations; see [16] for a discussion of the IEKF measurement update.

The image measurement equation combines the projection equations for all of the points visible in the current image i:

$$\begin{bmatrix} x_{0,u} \\ x_{0,v} \\ x_{1,u} \\ x_{1,v} \\ \vdots \\ x_{p-1,u} \\ x_{p-1,v} \end{bmatrix} = \begin{bmatrix} \pi_u(W_{\rho(\tau),t(\tau)}(X_{i,0})) \\ \pi_v(W_{\rho(\tau),t(\tau)}(X_{i,0})) \\ \pi_u(W_{\rho(\tau),t(\tau)}(X_{i,1})) \\ \pi_v(W_{\rho(\tau),t(\tau)}(X_{i,1})) \\ \vdots \\ \pi_u(W_{\rho(\tau),t(\tau)}(X_{i,p-1})) \\ \pi_v(W_{\rho(\tau),t(\tau)}(X_{i,p-1})) \end{bmatrix} + n_v \tag{4.20}$$

Here, $(x_{0,u}, x_{0,v}), (x_{1,u}, x_{1,v}), \ldots, (x_{p-1,u}, x_{p-1,v})$ are the projections visible in the current image i. As in Section 4.3, π is the projection from a three-dimensional, camera coordinate system point onto the image; $W_{\rho(\tau),t(\tau)}$ is the world-to-camera transformation specified by the Euler angles $\rho(\tau)$ and translation $t(\tau)$; and $X_{i,j}$ is the three-dimensional, world coordinate system position of the jth point in the current image. n_v is a vector of zero mean noise, which is normally taken to be isotropic with $\sigma = 2$ pixels.

The inertial measurement equation is:

$$\begin{bmatrix} \omega' \\ a' \end{bmatrix} = \begin{bmatrix} \omega - b_\omega \\ \Theta(\rho)^{\mathrm{T}}(a - g) - b_\alpha \end{bmatrix} + n_i \tag{4.21}$$

The top and bottom component equations of (4.21) are equivalent to (4.6) and (4.9), rearranged to given the biased angular velocity and biased linear acceleration. As before, ω' and a' are the camera system measurements from the rate gyro and accelerometer, respectively, and $\Theta(\rho)$ is the camera-to-world rotation specified by the Euler angles ρ. ρ, ω, b_ω, g, and b_a, and a are the same members of the state that we encountered in (4.11). n_i is a vector of Gaussian noise.

4.4.5 Newly acquired points

The algorithm generates an initial mean and covariance for the state estimate using the batch method described in Section 4.3. This method properly incorporates points that are seen at the beginning of the observation sequence into the estimate mean and covariance. To incorporate points that become visible after the batch initialization into the state estimate, we have to address three issues:

1. Determining which new points to incorporate into the state estimate.

2. Finding a mean for new points that are to be incorporated into the state estimate.

3. Finding a variance for new points that are to be incorporated into the state estimate, and the covariances that relate them to the estimates of the other state components and to the other new points.

This subsection describes the approach to each of these issues, with emphasis on the most difficult issue, correct computation of the variances and covariances.

To address the first issue, the incorporation of newly visible points into the state estimate is delayed until the point's three-dimensional position can be computed accurately relative to the camera positions from which it was visible, i.e., until its variance relative to the most recent camera position is below some threshold. This is to prevent numerical problems and reduce bias in the filter. The number of images required to achieve a sufficiently low variance in the point's position depends on the translation between the images.

Finding a mean for a point that has become visible after the batch initialization is straightforward. The camera positions estimated by the filter for the images in which the point was visible are used to triangulate the point's position.

Determining the new state covariance matrix is more complex, and must consider the following interrelated requirements:

1. The mean and covariance estimates produced by the recursive estimation should approximate those produced by a batch estimation as closely as allowed by the choice of the filter state and the filter's Gaussian prior assumption. This requirement is a fundamental design goal of recursive estimation algorithms, and requirements 2, 3, and 4 below are satisfied if this requirement is satisfied.

2. The state estimate distribution (i.e., mean and covariance) that results from the point initialization should not be too optimistic, which would weight the observations used in the point initialization more than those used in subsequent measurement updates, or too conservative, which would weight the observations used in the point initialization less than those used in subsequent measurement updates. This is important to ensure that the resulting estimates will be consistent with all of the observations, insofar as the observations allow. This also ensures that the available observations are best utilized, which is important in sequences where each point is only visible in a short subsequence of the image sequence.

3. The resulting filter should accurately estimate the covariances of the camera position, camera velocity, and point positions in the world coordinate system so that these can be monitored in critical applications. This is particularly important for long sequences, in which these variances will necessarily increase with time as points enter and leave the camera's field of view.

4. The variance of points relative to the current camera position, which is generally lower than the variance of the points relative to the world coordinate system, should be accurately estimated. This is important if the output of the filter will be used to generate image coordinate system priors for tracking the points in subsequent images.

To satisfy these requirements as closely as possible, the method described by Smith, *et al.*[54] for integrating newly visible landmarks into extended Kalman filters in simultaneous localization and mapping (SLAM) applications has been adapted. The motivation of Smith, *et al.*'s method is to properly incorporate sensor coordinate system measurements and covariances into global coordinate system estimates and covariances given the covariance associated with the sensor position, and to allow the sensor coordinate system measurement and covariance to be reextracted from the global coordinate system measurement and covariance. As described in subsection 2.4, SLAM applications typically use active range sensors, which provide measurements of both the range and bearing (i.e., the point's full three-dimensional location) in the sensor's coordinate system. In motion estimation from image measurements, only the bearing is available from any one sensor measurement. The rest of this subsection briefly describes Smith, *et al.*'s method, describes how it has been applied to the bearings-only case, and discusses the relative advantages and disadvantages of this approach versus some possible alternative approaches.

Smith, *et al.*'s method operates as follows. Suppose that:

- x is the filter's state estimate before the incorporation of the new point estimate, and that $C(x)$ is the covariance estimate for x produced by the most recent measurement update step
- z_w is the world coordinate system point corresponding to the sensor coordinate system point z_c
- $g(x, z_c)$ is the rigid transformation that maps z_c to z_w
- G_x and G_{z_c} are the derivatives of g with respect to x and z_c, respectively, evaluated at the current estimates of x and z_c

Then, Smith *et al.*'s method transforms the sensor coordinate system covariance $C(z_c)$ into a world coordinate system covariance $C(z_w)$, and establishes a cross-covariance $C(x, z_w)$ between x and z_w, using

$$C(z_w) = G_x C(x) G_x^{\mathrm{T}} + G_{z_c} C(z_c) G_{z_c}^{\mathrm{T}} \tag{4.22}$$

$$C(x, z_w) = G_x C(x) \tag{4.23}$$

The new point can then be incorporated into the state estimate by augmenting x with z_w, and $C(x)$ with $C(z_w)$ and $C(x, z_w)$.

To adapt this method to the bearings-only case by assuming that the camera position estimates corresponding to the initialization images in which the point was visible are correct with respect to each other. Mean and covariance estimates for the point's position relative to the most recent camera position can be computed under this assumption, effectively producing a "virtual range device" that can be used in Smith, *et al.*'s method. The mean and variance of this virtual device's position with respect to the world coordinate system are that of the most recent camera position.

This method correctly accounts for the variances in the image observations used in the point initialization and the estimated uncertainty in the current camera estimate, but not for any relative error between the recent estimated camera positions. As a result, this method produces covariances that are quite close to those produced by a batch estimation if the relative error between the camera estimates is small, satisfying requirement 1 and requirements 2, 3, and 4, which follow from it. However, in those cases where there is a significant relative error between the recent camera estimates (e.g., because of point feature mistracking), the new point will be initialized with an overly optimistic covariance, and subsequent state estimates will be contaminated by this error.

This approach is superior to two alternatives. In the first, camera system variances for the newly visible points are computed using the error variances associated with the points' image feature observations, and rotated into the world coordinate system using the current camera-to-world rotation estimate. This strategy correctly reflects the accuracy of the points' estimated positions relative to the current camera position (requirement 4), but does not capture the potentially high error in the points' estimated positions relative to the world coordinate system (requirement 3). This potentially large error in the points' world coordinate system position estimates is inevitable, since the error in the camera positions used to estimate the features' positions will grow large for long image sequences. After these inadvertently low variances have been assigned to the points, all of the filter's error variances will become corrupted, since subsequent camera positions will be estimated with respect to points that appear to have a low world coordinate system variances, and will therefore be assigned low variances in turn (violating requirement 1).

In the second, fixed variances, the variances described in the previous paragraph, or some other reasonable camera coordinate system variances are adopted, but inflated in an attempt to reflect the potential inaccuracy in the points' estimated positions relative to the world coordinate system. This strategy is poor for the following reasons. First, this strategy does not reflect the accuracy of the points' estimated positions relative to the current camera position, which may be useful information, e.g., for tracking the point's position in subsequent images (requirement 4). Second, for any fixed inflation factor, this choice also may not reflect the point's inaccuracy in the world coordinate system, which may grow arbitrarily large given long sequences (requirement 3). Third, this method effectively jettisons all of the information about the points' positions provided by the points' initial image feature observation variances, so the estimated three-dimensional point positions may become inconsistent with the initial views of the point (requirement 2).

As mentioned above, this adaption of Smith, Self, and Cheeseman method cannot model the relative errors between the successive camera position estimates that comprise the "virtual range device". An alternative approach that models the relative errors between successive camera positions is the variable state dimension filter (VSDF) described by McLauchlan[38]. The VSDF is a hybrid batch-recursive approach rather than a Kalman filter, so this modeling comes at the cost of increased computation. The batch method described in Section 4.3 could be adapted for hybrid batch-recursive operation within the VSDF framework. Within the Kalman filter framework described in this chapter, however, only the most recent camera position is included in the state. In this case, the adaptation of Smith, *et al.*'s method described in this section is recommended.

4.4.6 Lost points

To limit the filter state dimension as new points become visible and are added to the state, the recursive method removes the three-dimensional positions of points lost by the tracker from the state by removing the corresponding entries of the mean and covariance estimates, and leaving all other mean and covariance values unchanged. This is equivalent to marginalizing over the removed point's state components.

4.4.7 Image-only estimation

Section 5.6 describes both recursive image-and-inertial estimates and recursive image-only estimates. The recursive image-only algorithm is similar to the recursive image-and-inertial algorithm described in this section, with the following differences:

- Only the camera rotation, the camera translation, and the three-dimensional positions of the currently visible points are included in the state.
- The state propagation assumes slowly changing camera rotation and translation rather than slowly changing angular velocity and linear acceleration.
- The state estimate is initialized using the batch image-only algorithm rather than the batch image-and-inertial algorithm.
- There is no inertial measurement update.

The image measurement update step, the method for incorporating newly visible points, and the policy for dropping lost points from the state are the same as those for the recursive image-and-inertial algorithm.

4.4.8 Discussion

As mentioned in subsection 4.4.5, a common goal of recursive algorithm design is to incorporate measurements as they become available, while approximating the state mean and covariance estimates that would be produced by a batch algorithm given all of the measurements at once. For instance, the recursive image-only algorithm described in subsection 4.4.7 above produces estimates that closely match those of the batch image-only algorithm described in subsection 2.10. This is particularly true as the time propagation variances that specify how close we expect camera positions at adjacent times to be are allowed to grow, easing the assumption of motion smoothness, which the batch method does not incorporate.

The batch and recursive image-and-inertial algorithms described in this chapter are not as closely related in this way. For instance, the batch algorithm estimates the sensor position only at the time of each image, while the recursive algorithm estimates the sensor position at the time of each image and each inertial measurement.

More importantly, the multirate design of the recursive algorithm implicitly requires the assumption of smoothness in the angular velocity and linear acceleration across measurement times to exploit inertial and image measurements acquired at different times. Consider, for example, an inertial measurement followed by an image measurement. If the angular velocity and linear acceleration time propagation variances are high, as required in scenarios with erratic motion, then the state prior covariance that results from the time propagation between the inertial and image measurement times grows quickly and only loosely constrains the image measurement step estimate. So, the filter is free to choose an estimate at the time of the image measurement that is quite different than the state estimate that resulted from the inertial measurement step. On the other hand, motion smoothness is not a critical assumption for combining measurements taken at different image times, as it is in the recursive image-only filter, because these estimates are related through the three-dimensional positions of the points observed at both times.

The batch algorithm, including the integration functions I_ρ, I_v, and I_t that integrate inertial measurement between image times, neither requires nor exploits such an assumption, so the batch algorithm is stronger than the recursive algorithm in the presence of erratic motion. It appears that robustness to erratic motion is more valuable in practice than the assumption of motion smoothness, and Tomasi and Kanade[56] drew a similar conclusion in the context of shape-from-motion. Increasing the robustness of the recursive image-and-inertial algorithm to erratic motion is a promising direction for future work.

Chapter 5

Motion from image and inertial measurements: experiments

5.1 Overview

Chapters 3 and Chapter 4 above describe the potential advantages of motion from omnidirectional images and of motion from image and inertial measurements, respectively, and describe algorithms for exploiting these approaches. This chapter presents a suite of experiments in which sensor motion and other unknowns have been estimated using the algorithms described in those chapters, with emphasis on motion from image and inertial measurements. The experiments aim both to evaluate the specific algorithms that have been developed in those chapters, and to establish some fundamental facts about motion estimation using these modalities.

This chapter is organized as follows. Section 5.2 gives an overview of four suites of experiments. The camera configuration, observations, and results for each of these four suites are then described in detail in Sections 5.3-5.6.

As desribed in the introduction, motion estimation from omnidirectional images and motion from image and inertial measurements are two of the four approaches for improving motion estimation that this thesis investigates. Based in part on the results in this chapter, the conclusion in Chapter 8 contains some recommendations about the relative merits of, and for exploiting, these different approaches.

5.2 Experiments overview

This section gives a high level overview of the experiments (subsection 5.2.1), a description of the inertial sensor configuration (subsection 5.2.2), and the motion error metrics used to evaluate the accuracy of our estimates (subsection 5.2.3). The inertial sensor configuration and the motion error metrics are the same across the experiments described in the subsequent sections.

5.2.1 The experiments

A high level overview of the experiments is as follows:

Perspective arm experiment. The sensor rig includes a perspective camera and the inertial sensors and is mounted on a robotic arm that undergoes a known, preprogrammed motion. For each of the batch image-and-inertial, batch image-only, and recursive image-and-inertial algorithms, the estimation accuracy as the estimation parameters, the initial estimates, and the speed of the motion vary are explored.

Omnidirectional arm experiments. Three omnidirectional experiments were performing by mounting the two omnidirectional cameras and inertial sensors on the same arm used in the perspective arm experiment. Estimates were again generated using the batch image-and-inertial, batch image-only, and recursive image-and-inertial algorithms. In each experiment the motion is similar to that in the perspective arm experiment, so the relative merits of omnidirectional cameras and inertial sensors for motion estimation can be investigated.

Perspective crane experiment. The sensor rig includes a conventional camera and the inertial sensors, and is mounted on the platform of a robotic crane that can translate within a workspace of about 10 m $\times$ 10 m $\times$ 5 m. An estimate was generated using the recursive image-and-inertial algorithm.

Perspective rover experiments. The sensor rig includes a conventional camera and the inertial sensors, and is mounted on a ground rover that traverses approximately 230 meters. Estimates were generated using the recursive image-only algorithm and the recursive image-and-inertial algorithm.

Figure 5.1 shows the overall organization of the suite of experiments in more detail.

5.2.2 Inertial sensors

As described in the previous sections, a promising approach to a robust system for motion estimation is to augment motion estimation from image measurements with data from

PERSPECTIVE ARM EXPERIMENT (SECTION 5.3)

- Batch image−and−inertial estimates accuracy (subsection 5.3.3)
  - As the initial estimate and error function variances vary (set A, Table 5.1, Figures 5.3 and 5.4)
  - As the speed of the motion varies (set B, Table 5.2)
- Batch image−only estimates accuracy (subsection 5.3.4)
  - As the intrinsics calibration errors and image observation errors vary (Tables 5.3 and 5.4, Figure 5.5)
- Recursive image−and−inertial estimates accuracy (subsection 5.3.5)
  - As the time propagation variances vary (set A, Table 5.5, Figure 5.6)
  - As the number of initialization images varies (set B, Table 5.6)
  - As the speed of the motion varies (set C, Table 5.7)

OMNIDIRECTIONAL ARM EXPERIMENT (SECTION 5.4)

- For each of three experiments (subsection 5.4.2):
  - Experiment 1: low resolution, high speed, unmodeled observation errors
  - Experiment 2: high resolution, low speed
  - Experiment 3: low resolution, low speed
- consider (subsections 5.4.3, 5.4.4, 5.4.5, Table 5.8, Figure 5.9):
  - Batch image−and−inertial estimates accuracy
  - Batch image−only estimates accuracy
  - Recursive image−and−inertial estimates accuracy as the number of initialization images varies

PERSPECTIVE CRANE EXPERIMENT (SECTION 5.5)

- Recursive image−and−inertial estimate accuracy (subsection 5.5.3, Figures 5.11 and 5.12)

PERSPECTIVE ROVER EXPERIMENT (SECTION 5.6)

- Recursive image−only estimates accuracy
  - As the number of features in each image varies (subsection 5.6.3, Figure 5.15)
- Recursive image−and−inertial estimates accuracy
  - As the time propagation variances vary (subsection 5.6.4)

Figure 5.1: The organization of the experiments.

small, low-cost inertial sensors. To this end, the same configuration of inertial sensors has been used in all of the experiments, consisting of:

- Three single-axis, orthogonally mounted CRS04 rate gyros from Silicon Sensing Systems, which measure up to ± 150 degrees per second. Each of these gyros has approximate dimensions 1 cm $\times$ 3 cm $\times$ 3 cm, weighs 12 gm, and has a single unit cost of approximately $300.

- A Crossbow CXL04LP3 three degree of freedom accelerometer, which measures up to ± 4 g. This accelerometer has approximate dimensions 2 cm $\times$ 4.75 cm $\times$ 2.5 cm, weighs 46 gm, and has a single unit cost of approximately $300.

The voltages from the rate gyros and the accelerometer were captured at 200 Hz using two separate Crossbow CXLDK boards.

The three gyro voltage-to-rate calibrations were determined using a turntable with a known rotational rate. The accelerometer voltage-to-acceleration calibration was performed using a field calibration that accounts for non-orthogonality between the individual x, y, and z accelerometers. The gyro and accelerometer measurement error variances were found by sampling the sensors while they were kept stationary. The fixed gyro-to-camera and accelerometer-to-camera rotations differed between experiments but in each case were assumed known from the mechanical specifications of the mounts.

5.2.3 Motion error metrics

The simultaneous estimation of camera motion and scene structure from images alone recovers estimates only up to a scaled rigid transformation. That is, applying the same scaling and rigid transformation to all of the camera and point estimates produces new estimates that explain the observed data as well as the original estimates. With the incorporation of accelerometer data, it is possible to recover the global scale and two components of the absolute rotation, but with inexpensive accelerometers will likely be only roughly correct.

Therefore, the recovered estimates have been transformed into the ground truth coordinate system using a scaled rigid transformation before computing motion rotation and translation errors. This transformation has the form

$$t_g = sRt_e + t \tag{5.1}$$

where t_g and t_e are a camera translation in the ground truth and estimate coordinate systems, respectively, and s, R, and t are the scale, rotation, and translation of the scaled rigid transformation.

The scaled rigid transformation that best aligns the estimated camera translations with the ground truth translations, which can be found in closed form using the absolute orientation method described by Horn[28], has been used. An alternative is to align the estimate of the initial camera position with the ground truth initial camera position. However, this approach will not accurately capture the global accuracy of a motion estimate.

The reported rotation errors are the absolute rotation angle that aligns the estimated, transformed three-dimensional camera rotation with the ground truth rotation. This angle is the angle component of the angle-axis representation of the relative rotation. The reported translation errors are the distance in three-space between the estimated, transformed camera translation and the ground truth camera translations. For both rotation and translation, average and maximum errors are reported.

The error in global scale between the estimated and ground truth motion is reported. Specifically, the global scale is, as a percentage:

$$\text{scale error} = 1/s - 1 \tag{5.2}$$

where s is the transformation scale in (5.1). So, the scale error is greater than 0 if the scale of the estimated motion is larger than that of the ground truth motion, and less than 0 otherwise.

5.3 Perspective arm experiment

In the first experiment a perspective camera and the inertial sensors were mounted on a robotic arm to produce image and inertial observations from a known motion. The resulting observation sequence and ground truth motion information were used to investigate the relative performance of the batch image-and-inertial, batch image-only, and recursive image-and-inertial algorithms, as the estimation parameters, initial estimates, and the speed of the motion varied.

Subsections 5.3.1 and 5.3.2 below describe the sensor configuration and observations, respectively, for this experiment. Subsections 5.3.3, 5.3.4, and 5.3.5 describe the estimates produced by the batch image-and-inertial, batch image-only, and recursive image-and-inertial algorithms for from these observations. Because these sections include a large number of estimates, subsection 5.3.7 includes a concise summary of the key results.

5.3.1 Camera configuration

A conventional frame grabber was used to capture images at 30 Hz from a Sony XC-55 industrial vision camera paired with a 6 mm lens. The camera exposure time was set to 1/200 second to reduce motion blur. To remove the effects of interlacing, only one field was used from each image, producing 640 $\times$ 240 pixel images. As described in Section 2.9, the camera intrinsics model described by Heikkilä and Silvén[27] has been used. This calibration naturally accounts for the reduced geometry of our one-field images.

5.3.2 Observations

To perform experiments with known and repeatable motions, the rig was mounted on a Yaskawa Performer-MK3 robotic arm, which has a maximum speed of 3.33 meters per second and a payload of 2 kilograms. The programmed motion translates the camera x, y, and z through seven pre-specified control points, for a total distance traveled of about three meters. Projected onto the (x, y) plane, these points are located on a square, and the camera moves on a curved path between points, producing a clover-like pattern in (x, y). In z, the camera alternates 0.3 m between high and low positions at consecutive control points. The camera rotates through an angle of 270 degrees about the camera's optical axis during the course of the motion.

The observation sequence consists of 152 images and approximately 860 inertial readings, each consisting of both a gyro and accelerometer voltage. 23 features were tracked through the image sequence, but only 5 or 6 appear in any one image. Points were tracked using the Lucas-Kanade algorithm[37], but because the sequence contains repetitive texture and large interframe motions, mistracking was common and was corrected manually. Four example images from the sequence, with tracked points overlaid, are shown in Figure 5.2. In this experiment, as in all the following experiments, an isotropic observation error with variance $(2.0\,\text{pixels})^2$ for each tracked point position was assumed.

5.3.3 Batch image-and-inertial estimates

Two sets of estimates for the motion and other unknowns using the batch image-and-inertial algorithm were computed:

- Set A explores the accuracy of the motion estimates versus ground truth as the quality of the initial estimate and the variances that specify the inertial error function E_{inertial} vary.

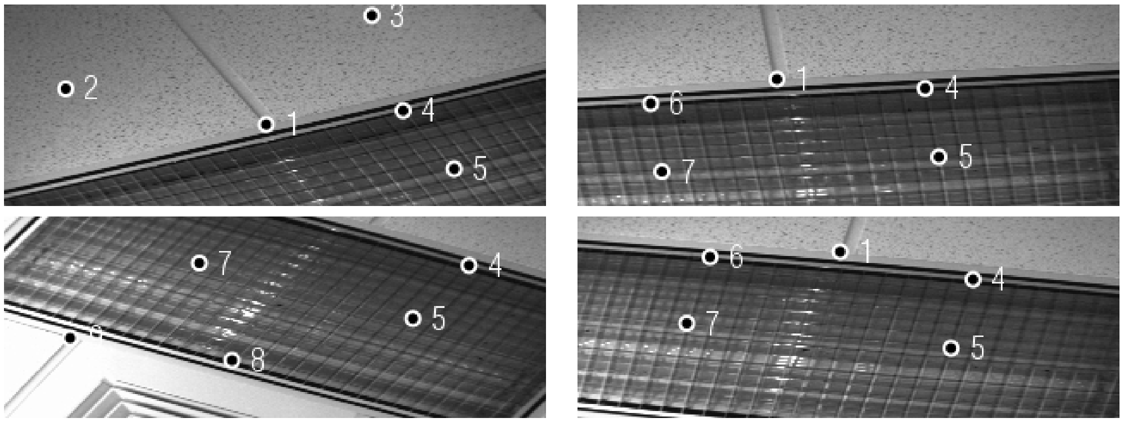


Figure 5.2: Images 16, 26, 36, and 46 from the 152 image sequence in the perspective arm experiment, with the tracked features overlaid, are shown clockwise from the upper left. As described in subsection 5.3.1, the images are one field of an interlaced image, so their height is half that of the full images.

- Set B explores the accuracy of the motion estimates versus ground truth as the speed of the motion and the variances that specify the inertial error function E_{inertial} vary.

This subsection describes the convergence of the batch algorithm and the error in the resulting estimates versus ground truth for each of these estimates.

Set A. For the estimates in Set A, the rotation, translation, and velocity error variances that specify the Mahalanobis distances D in E_{inertial} (described in subsection 4.3.2) were always chosen to be equal to each other, and varied together from 10^{-8} to 10^{-3}. For each of these inertial error term variance choices, the batch algorithm was run starting from each of four different initial estimates:

- An "accurate" estimate produced by the recursive image-and-inertial algorithm.

- A "poor" (i.e., qualitatively correct but not otherwise accurate) estimate produced by the recursive image-and-inertial algorithm.

- A "failed" (i.e., not qualitatively correct) estimate produced by the recursive image-and-inertial algorithm.

- A "blind" estimate produced using the default initialization method described in subsection 4.3.4. This initialization does not use the recursive method or any *a priori* knowledge of the unknowns.

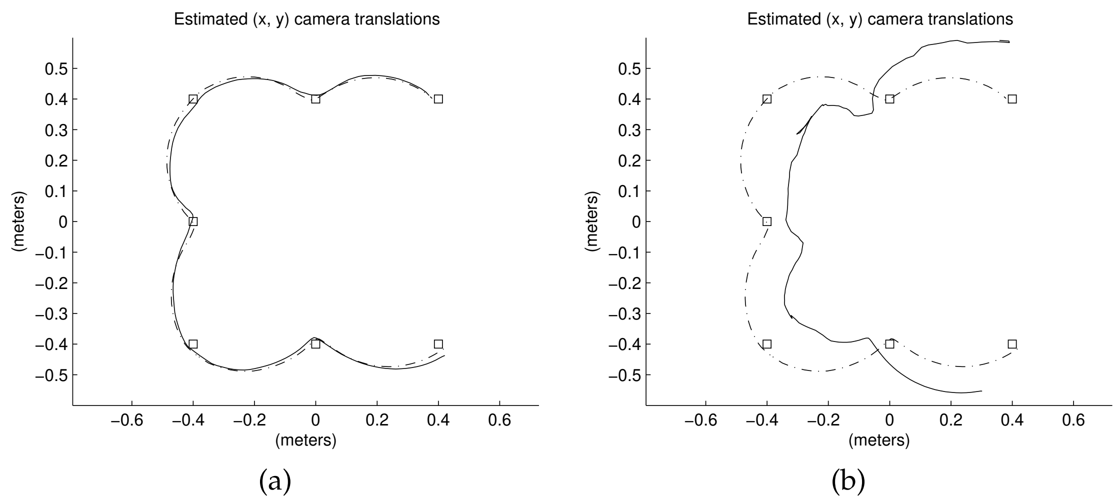


Figure 5.3: The x, y translation estimates generated by the batch image-and-inertial algorithm for the perspective arm experiment, assuming inertial error variances of 10^{-5}. Left: The final estimate and the accurate initial estimate are shown as the dash-dotted and solid lines, respectively. The x, y locations of the seven known ground truth points are shown as squares. Right: The final estimate and the poor initial estimate are shown as the dash-dotted and solid lines, respectively. The known ground truth points are again shown as squares.

The accurate, poor, and failed recursive estimates were generated using increasingly inappropriate choices of the recursive algorithm's angular velocity and linear acceleration propagation variances. The effect of varying these propagation variances on the recursive algorithm is considered in more detail in subsection 5.3.5 below, which is devoted to the accuracy of the recursive algorithm for this dataset.

Table 5.1 summarizes the results for Set A, including the number of iterations required for convergence and the error metrics for each combination of the E_{inertial} error function variances and initial estimate quality. For each choice of the inertial error variances, the algorithm converged in a few iterations from the accurate and poor initial estimates, and failed to converge from the failed initial estimate. Figure 5.3 shows the x, y translation estimate from the batch image-and-inertial algorithm assuming inertial error variances of 10^{-5}, along with the accurate and poor initial estimates used in the minimization.

The algorithm also converged from the blind initial estimate for most choices of the inertial error variances, but required more iterations than convergence from the accurate or poor recursive estimates. In those cases where the algorithm converged from the blind initial estimate, the convergence was in spite of a large difference between the blind estimate and the final estimate. For example, the camera positions in the final estimate span 270 degrees in rotation about z, whereas all of the camera positions in the blind initial estimate are zero. This is a much larger range of convergence than that found for batch image-only estimation by Szeliski and Kang[55], who report failure to converge for spans in rotation above 50 degrees.

The improved convergence above some inertial error variance threshold is an effect seen in many datasets, and presumably occurs because the inertial error function has a simpler topography than the image error function. In each case where the algorithm converged, it converged to the same estimates for a particular choice of the inertial error variances. In the table, this is reflected by the identical error metrics for each choice of the variances. Those cases where the method failed to converge from the blind estimate are indicated in the table by "N/A".

The rotation, translation, and scale errors versus ground truth were low across a wide range of the inertial error variances, and the trend in the translation errors as a function of the inertial error variances is shown in Figure 5.4. Variances between 10^{-8} and 10^{-4} all produced strong estimates. The estimates that result from the 10^{-3} variances, which weight the inertial measurements the least relative to the image measurements, are weak and show some of the problems inherent in estimating the motion from image measurements only. Subsection 5.3.4 below discusses the problems with image-only estimation for this data set.

Set B. As mentioned at the beginning of this subsection, a second set of estimates, Set B, which were produced by varying the speed of the motion and the inertial error term variances, was also considered. More specifically, the image and inertial observation sequences described in subsection 5.3.2 were artificially slowed by factors $f = 2$, $f = 4$, and $f = 8$ using the following four-step procedure:

1. Increase the image, gyro, and accelerometer timestamps by a factor of f.
2. Introduce $f - 1$ new interpolated inertial readings between each adjacent pair of original inertial measurements.
3. Reduce each of the observed gyro angular velocities by a factor of f and the observed accelerometer apparent accelerations by a factor of f^2.
4. Add Gaussian random noise to the scaled inertial measurements produced by step 3 to bring the measurement noise level back to that of the original observations.

This method works well for both the gyro and accelerometer measurements in this dataset, subject to the following caveats:

ρ, t, v variances	initial estimate	rotation error (rad)	translation error (cm)	scale error	iterations for convergence
10^{-8}	accurate recursive	0.13 / 0.20	3.3 / 5.3	-6.4%	3
	poor recursive	0.13 / 0.20	3.3 / 5.3	-6.4%	5
	blind	0.13 / 0.20	3.3 / 5.3	-6.4%	201
10^{-7}	accurate recursive	0.10 / 0.16	3.1 / 4.9	-6.0%	3
	poor recursive	0.10 / 0.16	3.1 / 4.9	-6.0%	5
	blind	0.10 / 0.16	3.1 / 4.9	-6.0%	163
10^{-6}	accurate recursive	0.09 / 0.14	2.6 / 3.9	-6.7%	3
	poor recursive	0.09 / 0.14	2.6 / 3.9	-6.7%	5
	blind	0.09 / 0.14	2.6 / 3.9	-6.7%	69
10^{-5} ($\star$)	accurate recursive	0.09 / 0.14	2.3 / 2.9	-8.2%	4
	poor recursive	0.09 / 0.14	2.3 / 2.9	-8.2%	5
	blind	0.09 / 0.14	2.3 / 2.9	-8.2%	209
10^{-4}	accurate recursive	0.09 / 0.12	2.7 / 3.7	-13.4%	5
	poor recursive	0.09 / 0.12	2.7 / 3.7	-13.4%	10
	blind	N/A	N/A	N/A	N/A
10^{-3}	accurate recursive	0.12 / 0.17	8.9 / 15.5	-29.3%	23
	poor recursive	0.12 / 0.17	8.9 / 15.5	-29.3%	23
	blind	N/A	N/A	N/A	N/A
image only	batch estimate ($\star$)	0.47 / 0.60	19.0 / 32.6	N/A	N/A

Table 5.1: The error metrics and iterations required for convergence for the Set A estimates from the perspective arm experiment. The rotation and translation error entries give the average error before the slash and the maximum error after the slash. The "N/A" entries indicate those cases where the algorithm failed to converge. The algorithm failed to converge from the failed recursive initial estimate for every choice of the inertial error variances, so these entries have been excluded from the table.

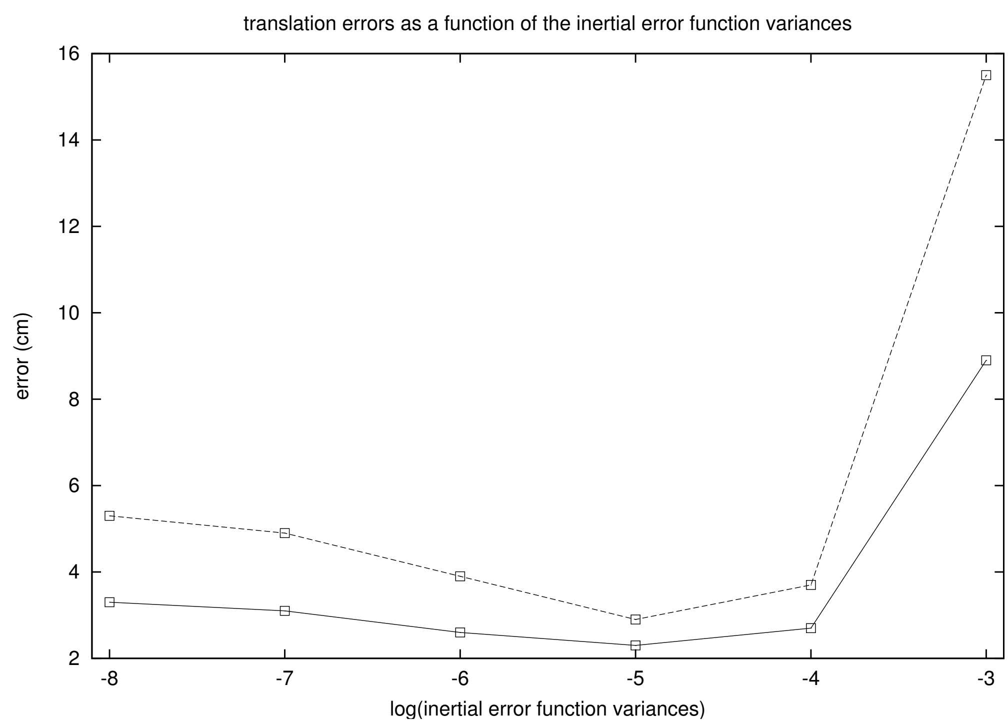


Figure 5.4: The average (solid line) and maximum (dashed line) translation errors from Table 5.1, shown as a function of the inertial error function variances. The change in the errors as a function of the inertial error function variances is small over a wide range of variances, from 10^{-8} to 10^{-4}. The same is true for the rotation and scale errors.

- The frequency of the image observations is reduced by a factor f as well as the motion speed.

- Since the effects of acceleration, accelerometer rotation, accelerometer bias, and gravity on the accelerometer measurements cannot be separated *a priori*, a simple scaling has been applied to both the gyro and accelerometer measurements in step 3 above. So, the effective bias and gravity in the resulting accelerometer measurements have magnitudes that are $1/f^2$ those of the original bias and gravity.

- The accelerometer measurements in this dataset contain high-frequency vibrations with magnitudes larger than the accelerometer noise level. So, interpolating smoothly between two measurements that result from vibration does not completely reflect the nature of the original motion.

For each inertial error variance between 10^{-6} and 10^{-4} and for each slowdown factor $f = 2, 4, 8$, the batch image-and-inertial algorithm was run from an initial estimate generated using the recursive image-and-inertial algorithm. The performance of the recursive image-and-inertial algorithm on these slowed observation sequences is described below in Section 5.3.5. The error metrics and number of iterations required for convergence for each of these estimates are given in Table 5.2. The quality of the estimates degrades and becomes more sensitive to the inertial variance choices as the slowdown factor increases, and one might expect this trend to continue as the slowdown factor increases further. However, the rate of degradation for the best error variance choice is reasonable.

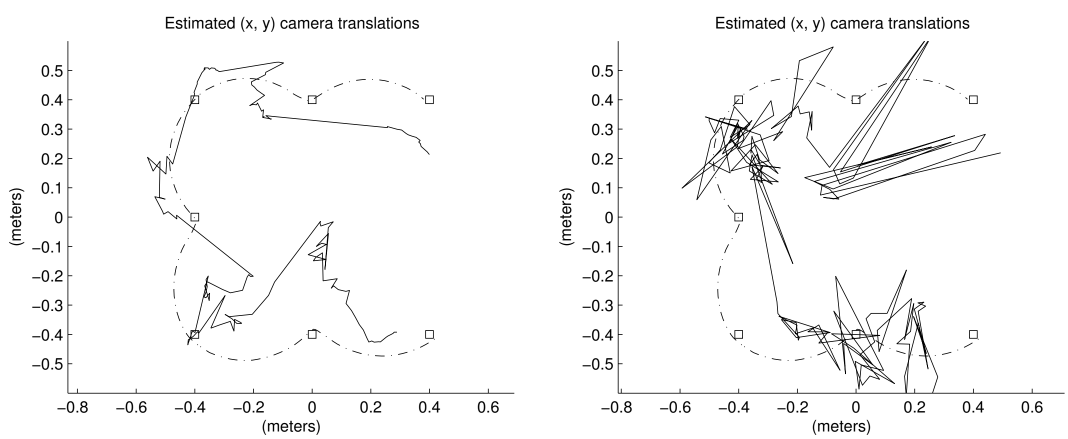
5.3.4 Batch image-only estimates

To highlight the complementary nature of the image and inertial observations, an estimate of the camera positions and three-dimensional point positions from image measurements only using the bundle adjustment algorithm described in subsection 2.10 was generated. The initial estimate used was the batch image-and-inertial estimate marked ($\star$) in Table 5.1. The rotation and translation errors for the image-only estimate are given in the last row of Table 5.1. No scale error is reported for this estimate because, as mentioned in subsection 5.2.3, no global scale is recovered by image-only motion estimation. The x, y translations estimated using the image-only method are shown with the image-and-inertial translation estimates in Figure 5.5(a).

The motion estimate from image measurements has large errors, and the errors versus ground truth are much higher than for the image-and-inertial estimate. The image-only algorithm reduces the image reprojection error versus the batch image-and-inertial estimate,

slowdown factor	ρ, t, v variances	rotation error (radians)	translation error (centimeters)	scale error	iterations for convergence
1	10^{-6}	0.09 / 0.14	2.6 / 3.9	-6.7%	3
	10^{-5}	0.09 / 0.14	2.3 / 2.9	-8.2%	4
	10^{-4}	0.09 / 0.12	2.7 / 3.7	-13.4%	5
2	10^{-6}	0.09 / 0.12	2.4 / 3.3	-9.1%	4
	10^{-5}	0.11 / 0.17	2.3 / 3.0	-14.9%	4
	10^{-4}	0.13 / 0.18	3.4 / 5.4	-19.7%	6
4	10^{-6}	0.13 / 0.19	2.7 / 3.3	-14.1%	7
	10^{-5}	0.24 / 0.30	5.6 / 8.4	-29.0%	16
	10^{-4}	0.37 / 0.41	8.8 / 12.7	-37.0%	30
8	10^{-6}	0.17 / 0.25	4.5 / 6.7	-22.2%	59
	10^{-5}	0.24 / 0.33	7.4 / 11.3	-34.8%	8
	10^{-4}	0.43 / 0.47	10.7 / 14.6	-44.9%	116

Table 5.2: The error metrics and iterations required for convergence for the Set B estimates from the perspective arm experiment. As in Table 5.1, the rotation and translation error entries give the average error before the slash and maximum error after the slash.



(a) (b)

Figure 5.5: Estimates for the perspective arm experiment generated using the batch image-only method. Left: The batch image-and-inertial and batch image-only x, y translation estimates are shown as the dash-dotted and solid lines, respectively. Right: The batch image-and-inertial estimate, and the batch image-only estimate generated from synthetic projections perturbed by isotropic, $(2.0\,\mathrm{pixel})^2$ variance noise are shown as the dash-dotted and solid lines.

camera intrinsics parameter	estimated value, standard deviation
x, y focal lengths in pixels	[834.18 419.88] $\pm$ [3.20 1.82]
x, y image center in pixels	[317.34 105.30] $\pm$ [9.46 4.34]
2nd, 4th order radial distortion coefficients	[-0.290 0.557] $\pm$ [0.030 .210]
y, x tangential distortion coefficients	[-.00621 -0.00066] $\pm$ [0.0024 0.0015]

Table 5.3: The camera intrinsics estimates and standard deviations for the perspective arm experiment, which were used to generate perturbed intrinsics for the experiments in subsection 5.3.4.

but in this case, reducing the image reprojection error degrades rather than improves the motion estimate.

What is the source of the large estimation errors produced by the batch image-only method? When synthetic, zero error image measurements are generated from the batch image-and-inertial camera and three-dimensional point estimates, and the batch image-only algorithm is applied to a perturbed version of the batch image-and-inertial estimate, the batch image-and-inertial estimate is correctly recovered from the image data alone. This is reflected in row 1 of Table 5.4. It follows that, while the shape and motion in this example are not strictly degenerate, estimating the motion from image observations only for this dataset is highly sensitive to errors in either the image observations or the camera intrinsics calibration.

To investigate the relative effects of image observation errors and camera intrinsics calibration errors, 10 additional estimates from the synthetic observations described above have been generated. In the first five, synthetic, zero-error image observations were used, but the camera intrinsics used in the estimation were randomly perturbed according to the standard deviations for these values reported by the camera intrinsics calibration program. The estimated camera intrinsics, along with the reported standard deviations, are given in Table 5.3.

The results are shown in rows 2-6 of Table 5.4. The resulting errors in the estimates are on the order of, or less, than the errors that observed in the batch image-and-inertial estimates from the real image measurements.

In the second five experiments, unperturbed camera intrinsics values were used, but noise was added to the synthetic projections with a standard deviation of 2.0 pixels in each direction, which is the same observation error distribution assumed in the experiments. The resulting errors are shown in rows 7-11 of Table 5.4, and are an order of magnitude larger than the errors that result from perturbing the camera intrinsics. So, the sensitivity

trial	perturbed instrinsics	perturbed projections	rotation error (radians)	translation error (centimeters)
1	no	no	3.4×10^{-6} / 1.1×10^{-5}	3.3×10^{-6} / 9.6×10^{-6}
2	yes	no	0.032 / 0.056	1.0 / 2.5
3	yes	no	0.030 / 0.037	1.3 / 2.7
4	yes	no	0.094 / 0.130	3.0 / 8.0
5	yes	no	0.046 / 0.070	1.5 / 4.0
6	yes	no	0.028 / 0.043	1.3 / 2.8
7	no	yes	0.27 / 0.60	22.9 / 60.1
8	no	yes	0.19 / 0.57	21.0 / 53.2
9	no	yes	0.42 / 1.07	34.4 / 70.3
10	no	yes	0.57 / 0.90	29.8 / 59.2
11	no	yes	0.32 / 0.66	17.7 / 61.5

Table 5.4: The sensitivity of image-only batch estimation to camera intrinsics calibration errors and to image observation errors. The correct motion can be recovered from image measurements only given synthetic, zero noise observations and the intrinsics used to generate them (row 1). Estimating the motion from the synthetic image observations and a perturbed version of the camera intrinsics that generates the observations results in errors (rows 2-6) are much less than the errors that result from estimating the motion from the unperturbed camera intrinsics and noisy image observations (rows 7-11).

to image observation errors is a major source of error in the image-only motion estimates for this dataset. Furthermore, the image observation errors are a larger source of error in the estimated motion than error in the camera intrinsics calibration, unless the intrinsics calibration algorithm grossly underestimates the intrinsics error variances.

5.3.5 Recursive image-and-inertial estimates

Three sets of estimates from the perspective arm observations were computed using the recursive image-and-inertial algorithm:

- Set A explores the estimation accuracy as the recursive algorithm's angular velocity and linear acceleration propagation variances vary.
- Set B explores the estimation accuracy as the number of images used in the recursive algorithm's batch initialization varies.
- Set C explores the estimation accuracy as the speed of the motion varies.

Set A. 25 estimates were computed using the recursive image-and-inertial algorithm, and:

- varying the angular velocity propagation variance from $(10^{-2}\,\mathrm{radians/s})^2/\mathrm{s}$ to $(10^{6}\,\mathrm{radians/s})^2/\mathrm{s}$
- varying the linear acceleration propagation variance from $(10^{-2}\,\mathrm{m/s^2})^2/\mathrm{s}$ to $(10^{6}\,\mathrm{m/s^2})^2/\mathrm{s}$.

Guided by the batch algorithm results in Table 5.1, rotation, translation, and velocity error variances of 10^{-5} have been adopted for the batch initialization in each case, and the batch initialization uses 40 images in each case. The gyro and accelerometer measurements variances are from the sensor calibration described in subsection 5.2.2.

The results are summarized in Table 5.5 and Figure 5.6. Generally, the estimates are good if the angular velocity variance is 10^2 or less and if the linear acceleration is 10^0 or greater. If the linear acceleration is less than 10^0, then due to the strong accelerations in the motion, the filter is unable to track the changes in linear acceleration. Of the variances tests, a angular velocity variance of 10^{-2} and a linear acceleration variance of 10^4 provide the best estimate, so these propagation variances have been adopted in sets B and C below.

Set B. 5 estimates were computed using the recursive image-and-inertial algorithm varying the number of batch initialization images from 20 to 60. As in set A above, rotation, translation, and velocity error variances of 10^{-5} were used for the batch initialization, and based on the experiments in Set A above, angular velocity and linear acceleration propagation variances of 10^{-2} and 10^4 respectively have been used.

The results are summarized in Table 5.6. The quality of the estimates degrades as the number of initialization images goes down. In particular, initializing from only 20 images produces a poor recursive estimate for this dataset.

Set C. To investigate the robustness of the recursive method to changes in the overall speed of the motion, 3 additional estimates were computed using the recursive image-and-inertial algorithm on the artificially slowed observation sequences described in subsection 5.3.3 above. As in Set A, the batch initialization uses 40 images and 10^{-5} for the inertial error variances. Based on the results from Set A, 10^{-2} and 10^4 have again been adopted as the angular velocity and linear acceleration propagation variances, respectively.

The results are summarized in Table 5.7 below. As one might expect, the rotation and translation errors are slightly worse for the recursive algorithm than for the batch algorithm. However, their degradation as the overall motion speed and frequency of the image measurements decrease is reasonable, and comparable to that of the batch algorithm.

ω propagation variance	a propagation variance	rotation error (radians)	translation error (centimeters)	scale error
10^{-2}	10^{-2}	0.25	8.4	-3.0%
	10^{0}	0.12	4.2	-3.1%
	10^{2}	0.11	4.0	-2.8%
	10^{4}	0.10	3.4	0.6%
	10^{6}	0.12	3.5	-0.3%
10^{0}	10^{-2}	0.25	8.3	-1.3%
	10^{0}	0.13	4.2	-2.9%
	10^{2}	0.11	3.9	-2.6%
	10^{4}	0.10	3.5	0.9%
	10^{6}	0.12	3.6	-0.1%
10^{2}	10^{-2}	0.4	14.0	16.7%
	10^{0}	0.14	3.8	-4.3%
	10^{2}	0.13	3.5	-4.7%
	10^{4}	0.10	3.5	-0.07%
	10^{6}	0.11	4.0	1.3%
10^{4}	10^{-2}	N/A	N/A	N/A
	10^{0}	0.22	5.0	-8.1%
	10^{2}	0.20	5.1	-9.2%
	10^{4}	0.13	3.6	-3.0%
	10^{6}	0.26	10.7	-24.0%
10^{6}	10^{-2}	0.30	13.0	1.4%
	10^{0}	0.32	14.3	8.7%
	10^{2}	0.18	5.8	-7.6%
	10^{4}	0.13	4.0	-2.5%
	10^{6}	0.54	13.4	14.9%

Table 5.5: The average error metrics for the Set A estimates generated using the recursive image-and-inertial algorithm.

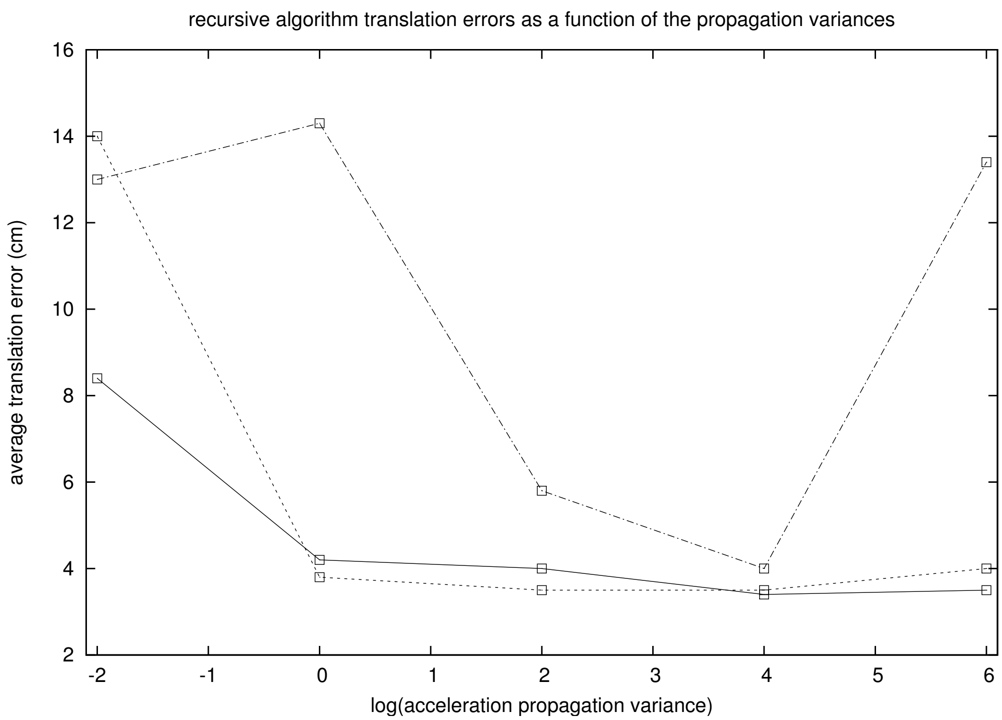


Figure 5.6: The average translation errors given in Table 5.5 for the recursive image-and-inertial algorithm, as a function of the linear acceleration propagation variance, for angular velocity propagation variances of 10^{-2} (solid line), 10^{2} (dashed line), and 10^{6} (dash-dotted line). The accuracy of the recursive method is dependent on choosing propagation variances appropriate for the sensors' motion.

initialization images	rotation error (radians)	translation error (centimeters)	scale error
20	1.21/1.39	15.7/42.2	-47.8%
30	0.22/0.26	5.7/6.9	20.0%
40	0.10/0.15	3.4/4.5	0.6%
50	0.10/0.16	2.7/3.7	-4.2%
60	0.08/0.12	2.1/2.8	-3.2%

Table 5.6: The error metrics for the Set B estimates generated using the recursive image-and-inertial algorithm. As in previous tables, the rotation and translation error entries give the average error before the slash and the maximum error after the slash.

slowdown factor	rotation error (radians)	translation error (centimeters)	scale error
1	0.10/0.15	3.4/4.5	0.6%
2	0.12/0.16	3.7/4.8	0.7%
4	0.17/0.21	3.2/5.3	-10.9%
8	0.13/0.16	5.4/8.0	-9.3%

Table 5.7: The error metrics for the Set C estimates generated using the recursive image-and-inertial algorithm. The rotation and translation error entries give the average error before the slash and the maximum error after the slash.

5.3.6 Covariance estimates

In the previous subsections, the effects of including gyro and accelerometer measurements on the accuracy of motion estimates for the perspective arm sequence were investigated by comparing the resulting estimates to the known ground truth points. Covariance estimates for the unknowns provide another measure of the benefits of including these measurements.

The inverse of the approximate Hessian matrix used by Levenberg-Marquardt in the batch method also provides an estimate for the unknowns' covariances. As shown in [6] and [50], the inverse of the approximate Hessian matrix is also the Cramer-Rao lower bound for the covariances in the case of maximum likelihood estimation with independent, isotropic Gaussian observation errors.

Covariances for the x and y translation estimates computed using this method are illustrated in Figure 5.7:

- Figure 5.7(a) shows the translation covariances computed assuming image measurements only and (2.0 pixel)^2 observation variances for image coordinate system projections. The 0.25 confidence level boundaries for every tenth camera position are shown as ellipses. Because the covariances are very large, 0.25 rather than 1.0 confidence level boundaries have been used for clarity.

- Figure 5.7(b) shows the translation covariances computed assuming image and gyro measurements, with (2.0 pixel)^2 observation variances for image coordinate system projections and 10^{-5} variances for the inertial error term's rotation components. The 1.0 confidence level boundaries for every tenth camera position are shown as ellipses.

- Figure 5.7(c) shows the translation covariances computed assuming image, gyro, and accelerometer measurements, with (2.0 pixel)^2 observation variances for image coordinate system projections and 10^{-5} variances for the inertial error term's rotation, velocity, and translation components. The 1.0 confidence level boundaries for every tenth camera position are shown as ellipses.

In each case, the covariances were computed using the "$\star$" estimate described in the Table 5.1. The increasing benefits of adding gyro and then accelerometer measurements are reflected in the covariance estimates.

5.3.7 Summary

The experiments in this section indicate:

1. The batch image-and-inertial algorithm has a wide range of convergence. In particular, the algorithm can converge from poor initial estimates, including estimates that use no *a priori* knowledge of the unknowns. However, the algorithm can be made to fail by choosing a severely inaccurate initial estimate. (Subsection 5.3.3, Set A.)

2. The batch image-and-inertial algorithm converges from a wider range of initial estimates than batch image-only estimation. (Subsection 5.3.3, Set A.)

3. The accuracy of the batch image-and-inertial algorithm is largely insensitive to the choices of the variances that define the inertial error term. (Subsection 5.3.3, Set A.)

4. The overall scale of the motion and scene structure can be recovered to within a few percent from image and inertial measurements, even when an inexpensive accelerometer not intended for navigation applications is employed (Subsection 5.3.3, Sets A and B).

5. The accuracy and convergence of the batch image-and-inertial algorithm degrade gracefully as the overall speed of the sensor motion decreases and the time between image measurements increases. (Subsection 5.3.3, Set B.)

6. Batch image-only estimation can be highly sensitive to errors in both the image measurements and the camera intrinsics calibration. However, for the dataset described in this section, the sensitivity to image measurements errors appears to be much higher. (Subsection 5.3.4.)

7. The recursive image-and-inertial algorithm can be highly sensitive to the angular velocity and linear acceleration propagation variances and to the number of initialization images, so these values need to be chosen in a way that accommodates the sensors' motion. (Subsection 5.3.5, Sets A and B.)

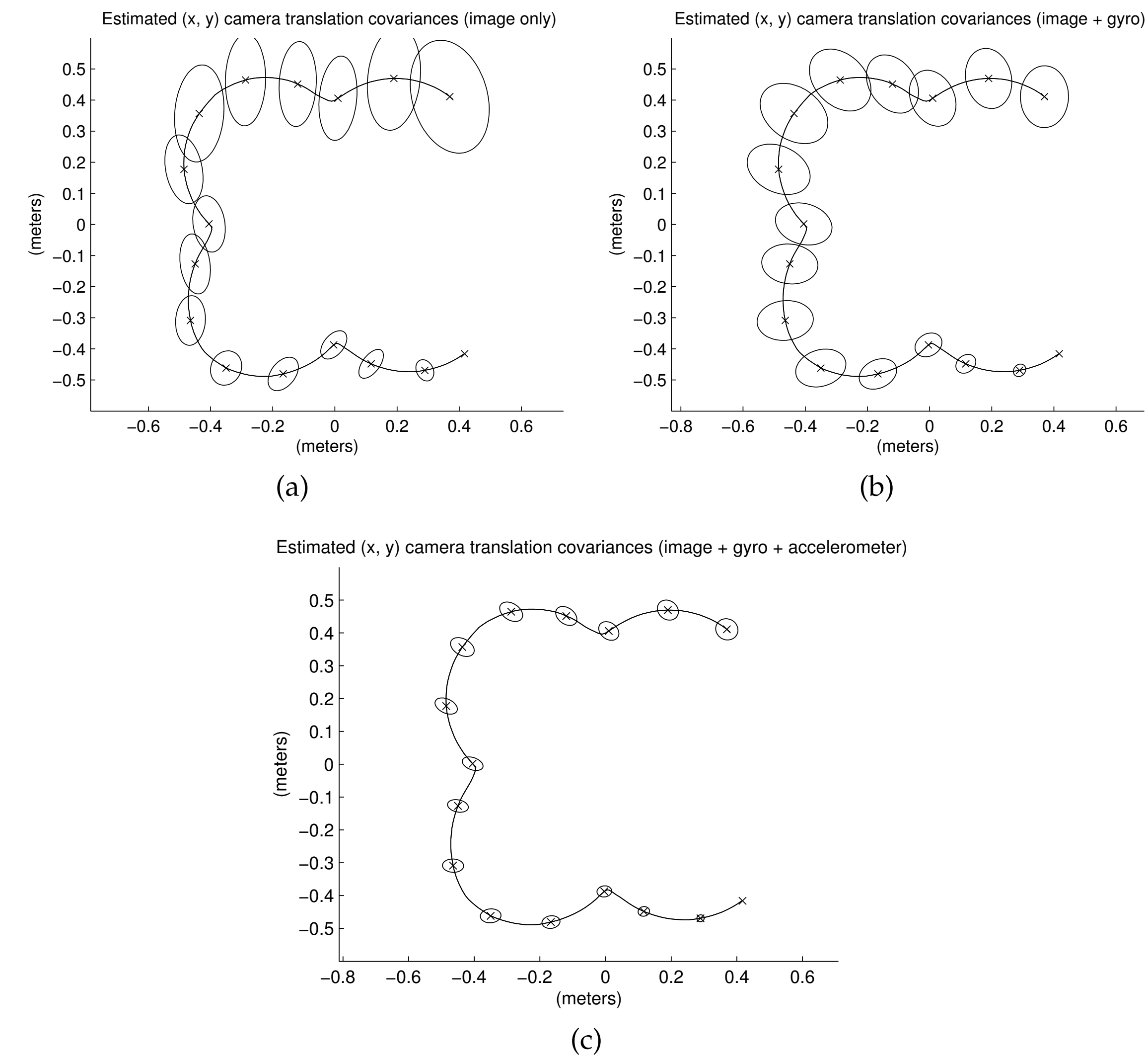


Figure 5.7: Translation covariances estimates for the perspective arm sequence: 0.25 confidence level boundaries assuming image measurements only (a); 1.0 confidence level boundaries assuming image and gyro measurements (b); 1.0 confidence level boundaries assuming image, gyro, and accelerometer measurements (c). The increasing benefits and adding gyro and then accelerometer measurements are reflected in the covariance estimates.

8. The sensitivity of the recursive algorithm to changes in the speed of the motion and the frequency of the image measurements is low, and is comparable to that of the batch algorithm. (Subsection 5.3.5, Set C.)

5.4 Omnidirectional arm experiments

To compare the relative benefits of using inertial measurements versus omnidirectional images, three experiments were conducted with the omnidirectional cameras shown in Figures 3.3 and 3.4, using the same arm used for the perspective experiment described in Section 5.3 above, and a similar motion. The sensor rigs and observation sequences for these experiments are described below in subsections 5.4.1 and 5.4.2, respectively. The batch image-and-inertial, batch image-only, and recursive image-and-inertial estimates are described in subsections 5.4.3, 5.4.4, and 5.4.5, respectively.

5.4.1 Sensor configuration

First omnidirectional configuration. The first omnidirectional camera configuration is shown in Figure 3.3 and consists of a CCD camera attached to a convex mirror by a rig that allows the relative rotation and translation between the mirror and camera to be manually adjusted. As in the perspective arm experiments described in the previous section, the CCD camera is a Sony XC-55, captured by a conventional frame grabber at 30 Hz, and the 640×480, interlaced images are again reduced to 640×240, noninterlaced images by removing one field. The camera was paired with a 16 mm lens for use with the mirror. The mirror is the equiangular mirror described by Ollis, *et al.*[46]. The adjustable rig is convenient, but as we'll see in subsections 5.4.2 and 5.4.4 below, is somewhat flexible and allows some unmodeled vibration between the camera and mirror during erratic motion.

Second omnidirectional configuration. The second omnidirectional configuration is shown in Figure 3.4 and consists of an IEEE 1394 camera attached to a convex mirror by a clear, rigid cylinder. The color, 1042×768 images produced by the camera were captured at the camera's maximum acquisition rate of 15 Hz, and were reduced to greyscale, 640×480 images before tracking. As with the first omnidirectional configuration, the camera was paired with a 16 mm lens. The mirror is again the equiangular mirror described by Ollis, *et al.*[46].

As described in Chapter 3, the motion estimation algorithms can incorporate knowledge of the six degree of freedom misalignment between the camera and mirror if this information is available, and an accurate estimate of this transformation generally increases

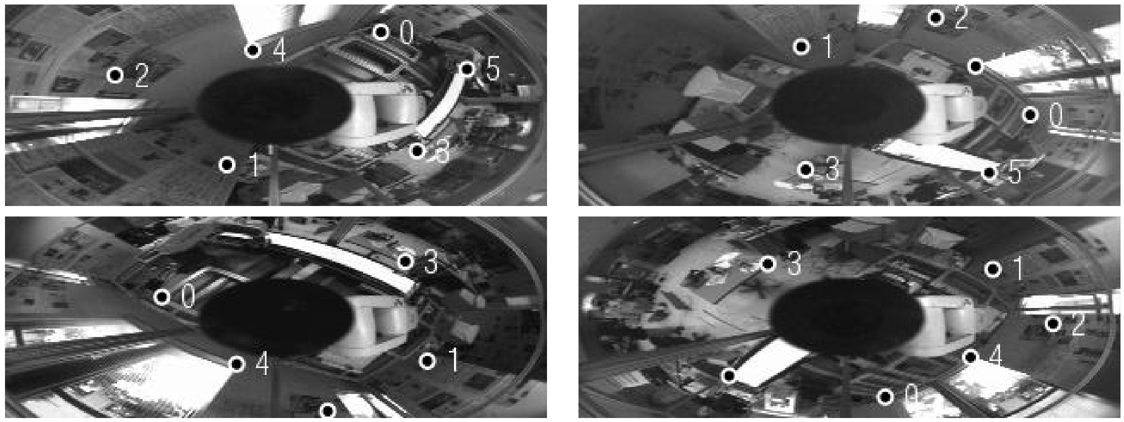


Figure 5.8: Images 0, 50, 100, and 150 from the 152 image sequence acquired with the first omnidirectional camera configuration for the first experiment, shown clockwise from the upper left. The tracked features are overlaid. As with the images in Figure 5.2, the images are one field of an interlaced image, so their height is one-half the height of the full images.

the accuracy of motion estimation from omnidirectional images. For the experiments described here, the precise mirror-to-camera transformations are not available, so reasonable but necessarily imprecise values for this transformation have been used in the experiments.

5.4.2 Observations

First experiment. In the first experiment, the first omnidirectional rig was mounted on the Performer arm and the same motion used for the perspective arm experiment, described in subsection 5.3.2, was executed. The resulting observation sequence contains 152 omnidirectional images and 862 inertial readings.

6 points were tracked through the sequence using Lucas-Kanade[37] with manual correction. In most of the images, all 6 points were visible. A few example images from this sequence, with tracked features overlaid, is shown in Figure 5.8. As mentioned above in subsection 5.4.1, the rig in the first configuration is somewhat flexible and allows some unmodeled vibration between the mirror and camera when the motion is erratic. So, some vibration of the mirror and of the tracked features, on the order of 10 pixels, is visible in the image sequence.

Second experiment. In the second experiment, the second omnidirectional rig was placed on the Performer arm, and moved through a motion similar to that used in for the perspective arm experiment and for the first omnidirectional experiment described above. Since the camera was able to acquire images at only 15 Hz rather than 30 Hz, the arm ve-

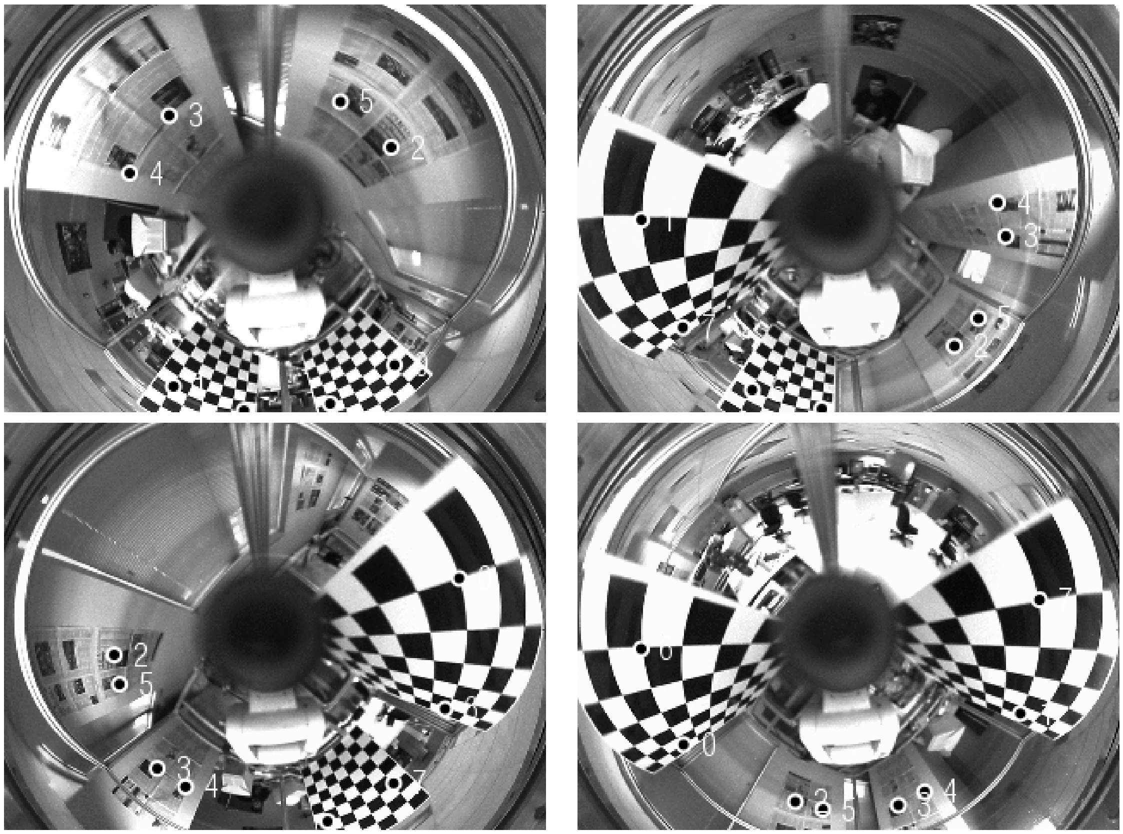


Figure 5.9: Images 0, 50, 100, and 150 from the 152 image sequence acquired with the second omnidirectional camera configuration for the second experiment, shown clockwise from the upper left. The tracked features are overlaid. In this experiment, the camera produces full resolution, noninterlaced images.

locity was reduced to half to produce an image sequence with the same number of images as the sequence used in the first experiment; inertial measurements were acquired at the same rate as in the previous experiments. The resulting observation sequence contains 152 omnidirectional images and 1853 inertial readings.

8 points were tracked through the entire sequence using Lucas-Kanade. Mistracking was again common and corrected by hand. Four example images from the sequence, with the tracked features overlaid, are shown in Figure 5.9.

Third experiment. For the third experiment, the image sequence from the second experiment was reduced to 640×240 to test the effects of image resolution on the omnidirectional motion estimates, and the eight points tracked in the second experiment were retracked through the entire sequence of reduced resolution images. The inertial measurements are the same as in the second experiment.

Comparisons with the perspective arm dataset. The motion speed and image resolution in the first omnidirectional experiment, which uses the first omnidirectional configuration, are the same as those used to produce the unslowed perspective arm estimates (the set A batch image-and-inertial estimates in subsection 5.3.3, the batch image-only estimates in subsection 5.3.4, and the set A and B recursive image-and-inertial estimates in subsection 5.3.5). So, the batch image-and-inertial, batch image-only, and recursive image-and-inertial estimates produced from this omnidirectional dataset can be compared against those described in Table 5.1, rows labeled "$\star$"; Table 5.1, row labeled "image only"; and Tables 5.5 and 5.6, respectively. One caveat is that, as we'll see below in subsection 5.4.4, the unmodeled vibration in the first omnidirectional experiment introduces local motion errors in the image-only estimates. Since this unmodeled vibration is not an inherent flaw in omnidirectional cameras, this should be considered when comparing the image-only estimates from the perspective arm experiment and the first omnidirectional arm experiment.

The motion speed and image resolution in the third omnidirectional experiment, which uses the second omnidirectional configuration and vertically reduced images, are the same as those used to produce the slowed perspective arm estimates with the slowdown factor $f = 2$ (the set B batch image-and-inertial estimates in subsection 5.3.3 and the set C recursive image-and-inertial estimates in subsection 5.3.5). So, the batch image-and-inertial and recursive image-and-inertial estimates from this omnidirectional experiment can be compared against those described in Table 5.2, $f = 2$; and Table 5.7, line 2, respectively.

5.4.3 Batch image-and-inertial estimates

For each of the three experiments a batch image-and-inertial estimate of the motion and other unknowns was generated, using an estimate from the recursive image-and-inertial method as the initial estimate; these recursive estimates are described below in subsection 5.4.5. Based on the results in subsection 5.3.3 exploring the accuracy of estimates as a function of the inertial error variances for the perspective sequence, inertial error variances of 10^{-5} have been used for each omnidirectional experiment.

The error statistics for each of the three batch image-and-inertial estimates are given in Table 5.8, and the x, y translation estimates from the first and second experiment are shown as the dash-dotted lines in Figure 5.10. In each experiment, the estimates suffer somewhat from the lack of a precise camera-to-mirror transformation calibration, but the results are comparable to those in the perspective arm experiment, despite the changes in motion speed, frame rate, and image resolution across the three experiments.

5.4.4 Batch image-only estimates

For each of the three experiments, a batch image-only estimate was generated. For consistency with the perspective experiments described in subsection 5.3.4, the batch image-and-inertial estimate has been used as the initial estimate for the image-only initialization. The error statistics for each of the image-only estimates are also given in Table 5.8, and the x, y translation estimates for the first and second experiments are shown as the solid lines in Figure 5.10. Unlike the perspective image-only estimate, each of the three omnidirectional estimates is globally correct, which reflects the advantage of using omnidirectional cameras over perspective cameras for motion estimation. However, as mentioned in subsection 5.4.1, the first omnidirectional camera configuration, which is used in the first experiment, suffers from an unmodeled vibration between the camera and mirror when the camera's motion is erratic. The effects of this vibration appear in the batch image-only estimate on the left of Figure 5.10 as local perturbations in the motion. Interestingly, the use of inertial measurements in the batch image-and-inertial estimate eliminates these perturbations.

5.4.5 Recursive image-and-inertial estimates

A sequence of estimates was also generated using the recursive image-and-inertial algorithm, varying the number of images used in the batch initialization. Based on the experiments described in subsection 5.3.3 and 5.3.5, inertial error variances of 10^{-5} for the batch initialization, 40 images in the batch initialization, and angular velocity and linear acceleration propagation variances of 10^{-2} and 10^{4} for recursive operation have been used.

The error metrics for the recursive estimates are also given in Table 5.8. In the first experiment, which uses the adjustable omnidirectional camera rig, the recursive algorithm is unable to overcome the unmodeled effects of vibration between the camera and mirror caused by the erratic motion of the arm, whereas the batch image-and-inertial method is robust to this problem. The recursive method produces accurate estimates for the second and third experiments, which use the more rigid second omnidirectional camera rig, and for those experiments the estimates degrade gracefully as the number of initialization images is decreased.

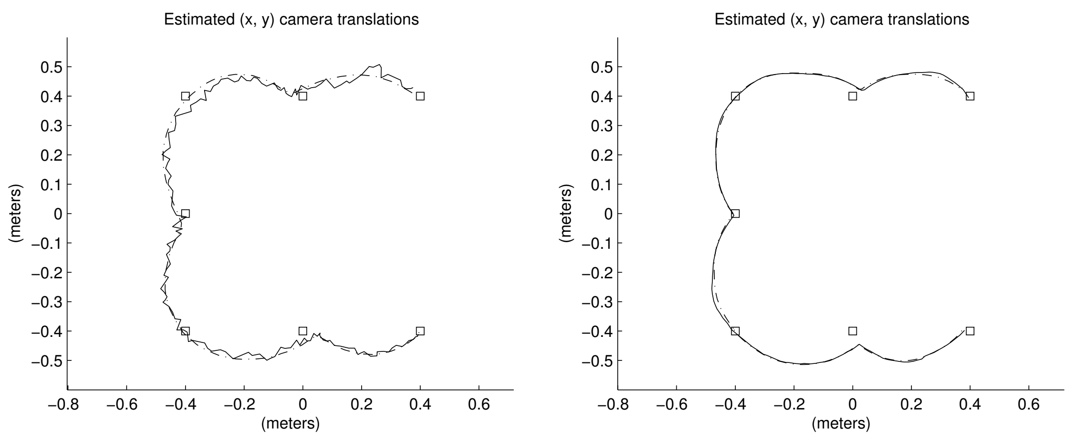
5.4.6 Summary

The experiments in this section indicate:

1. Image-only motion estimation from omnidirectional images is less sensitive to random image measurement errors than image-only motion estimation from perspec-

experiment	estimate	rotation error (radians)	translation error (centimeters)	scale error
first	batch image-and-inertial	0.12 / 0.15	3.4 / 4.5	5.2%
	batch image-only	0.12 / 0.15	5.1 / 7.5	N/A
	recursive, 20 image init.	0.14 / 0.18	7.4 / 12.3	-33.8%
	recursive, 30 image init.	0.12 / 0.15	7.5 / 14.2	-48.9%
	recursive, 40 image init.	0.12 / 0.20	9.6 / 15.6	-6.1%
	recursive, 50 image init.	0.11 / 0.15	6.4 / 9.2	-5.8%
	recursive, 60 image init.	0.11 / 0.15	6.1 / 8.1	2.8%
second	batch image-and-inertial	0.11 / 0.13	4.1 / 6.3	6.7%
	batch image-only	0.10 / 0.12	4.0 / 6.1	N/A
	recursive, 20 image init.	0.14 / 0.22	6.2 / 9.5	113.5%
	recursive, 30 image init.	0.14 / 0.18	4.7 / 7.0	-11.6%
	recursive, 40 image init.	0.13 / 0.17	4.1 / 5.2	-13.3%
	recursive, 50 image init.	0.13 / 0.17	3.9 / 5.2	-9.2%
	recursive, 60 image init.	0.11 / 0.15	3.9 / 4.9	8.5%
third	batch image-and-inertial	0.11 / 0.14	4.0 / 5.9	8.0%
	batch image-only	0.11 / 0.14	4.0 / 5.9	N/A
	recursive, 20 image init.	0.15 / 0.20	4.9 / 8.3	41.5%
	recursive, 30 image init.	0.15 / 0.19	4.6 / 7.1	-14.6%
	recursive, 40 image init.	0.14 / 0.17	4.3 / 5.5	-13.9%
	recursive, 50 image init.	0.14 / 0.17	4.2 / 5.4	-10.5%
	recursive, 60 image init.	0.12 / 0.15	4.0 / 4.8	9.4%

Table 5.8: The error metrics for the first, second, and third omnidirectional arm experiments. The estimates are largely insensitive to the changes in the speed and image resolution across the three experiments, and the sensitivity of the batch image-only estimates apparent in the perspective arm experiment is eliminated by use of the omnidirectional camera.



(a) (b)

Figure 5.10: The x, y translation estimates generated by the batch image-and-inertial and batch image-only algorithms for the first and second omnidirectional arm experiments. Left: The estimates generated by the batch image-and-inertial algorithm and by the batch image-only algorithm for the first omnidirectional experiment are shown as the dash-dotted and solid lines, respectively. Right: The estimates generated by the batch image-and-inertial algorithm and by the batch image-only algorithm for the second omnidirectional experiment are shown as the dash-dotted and solid lines, respectively. On both the left and right, the x, y locations of the seven known ground truth points are shown as squares.

tive images, so the advantage of incorporating inertial measurements is less for omnidirectional images than for perspective images. (Subsections 5.4.3 and 5.4.4.)

2. The incorporation of inertial measurements in the batch image-and-inertial algorithm provides some robustness to moderate, nonrandom image measurement errors, in this case unmodeled vibration between the mirror and camera in the first experiment. (Subsections 5.4.3 and 5.4.4.)

3. The recursive image-and-inertial algorithm produces poor estimates in the presence of nonrandom image measurement errors in the first experiment, despite the incorporation of inertial measurements. (Subsection 5.4.5.)

5.5 Perspective crane experiment

5.5.1 Camera configuration

The camera configuration for the perspective crane experiment is the same as that for perspective arm experiment, except that to gain a wider field of view, a 3.5 mm lens was used in place of the 6 mm lens.

5.5.2 Observations

The rig was mounted on the platform of a specially constructed crane that translates the platform within a 10 m $\times$ 10 m $\times$ 5 m area. For the experiment, the crane was programmed to move between eleven prespecified control points. These points take the platform from the center of the work area to one corner of a 6 meter wide square aligned with the crane coordinate system (x, y), around the square, and then back to the center of the work area. In z, the platform translates 1 m between adjacent control points, alternating between high and low positions.

The observation sequence taken during this circuit consists of 1430 images and 51553 inertial readings spanning 4 minutes and 46 seconds; eight example images from the sequence are shown in Figure 5.11. 100 points were extracted in the first image of the sequence, and tracked through the sequence using Lucas-Kanade. In subsequent images in which the number of tracked points fell below 80 due to features leaving the field of view, becoming occluded, or changing appearance, reextraction was used to bring the number of tracked points back to 100. Many mistracked points were identified by applying RANSAC in conjunction with batch, image-only shape and motion estimation to each subsequence

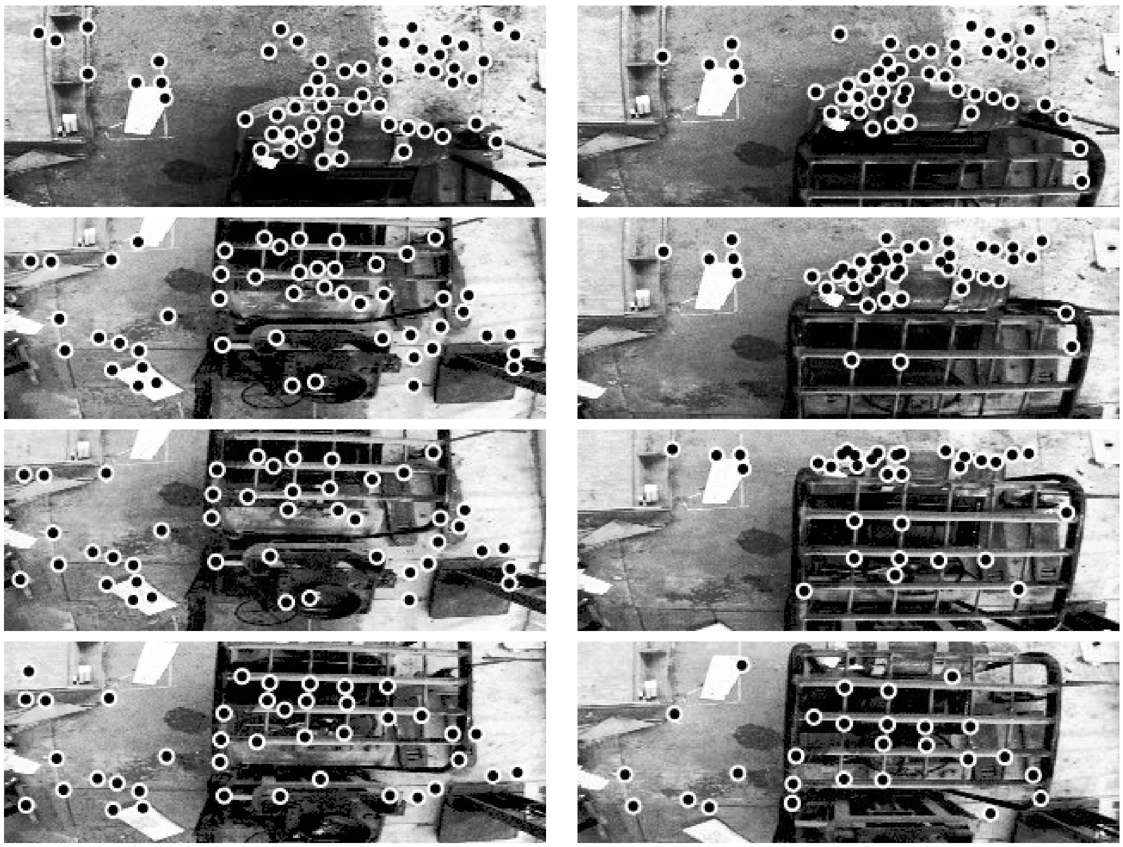


Figure 5.11: Images 480, 490, 500, . . ., 550 from the 1430 image sequence in perspective crane experiment are shown clockwise from the upper left, with the tracked image features overlaid. As with the images from the perspective arm experiment shown in Figure 5.2, the images are one field of an interlaced image, so their height is half that of a full image.

of 30 images in the sequence. 166 additional point features that were mistracked but not detected by the RANSAC procedure were pruned by hand.

These tracking and pruning stages resulted in a total of 1243 tracked points. Each image contains an average of 53.8 points, and each point is visible in an average of 61.9 images, so that on average each point was visible in 4.3% of the image sequence. As described in subsection 2.4, this is a very low percentage by the standards of image-only motion estimation.

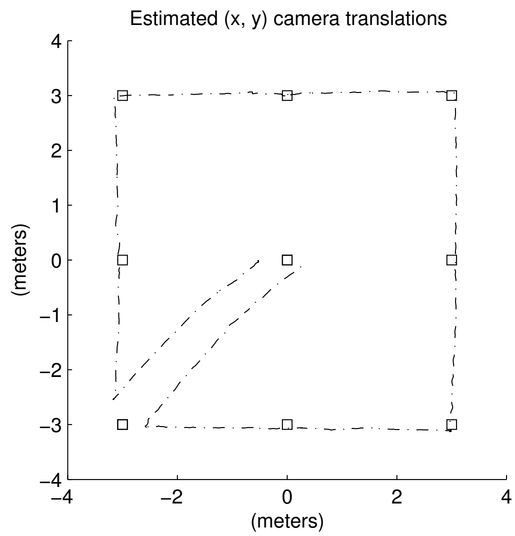


Figure 5.12: The x, y translation estimate for the perspective crane experiment generated by the recursive image-and-inertial method is shown by the line. The x, y locations of the eleven known ground truth points are shown as squares. Ground truth points 0 and 1 are identical to points 10 and 9, respectively, so that only 9 squares are visible in the figure.

5.5.3 Estimate

An estimate of the motion and other unknowns was generated with the recursive image-and-inertial method described in Section 4.4, using 40 images in the batch initialization. In this experiment, the average (maximum) rotation and translation errors are 13.6 (0.17) radians and 31.5 (61.9) cm, respectively. This average translation error is approximately 0.9% of the total 34.1 m traveled. The scale error in this experiment is -3.4%.

The resulting x, y translation estimates are shown from above in Figure 5.12, and the resulting z translation estimates are shown in Figure 5.13. In each figure the squares are the known ground control points.

5.6 Perspective rover experiment

The Hyperion rover is a test bed for candidate technologies for a long-range, robotic search for life on Mars. During April 2003, Hyperion performed the first of three scheduled field tests in Chile's Atacama Desert, and on eight of those days, Hyperion carried the camera, gyro, and accelerometer used in our experiments. This section describes the observations and estimates for one of the resulting observation sequences.

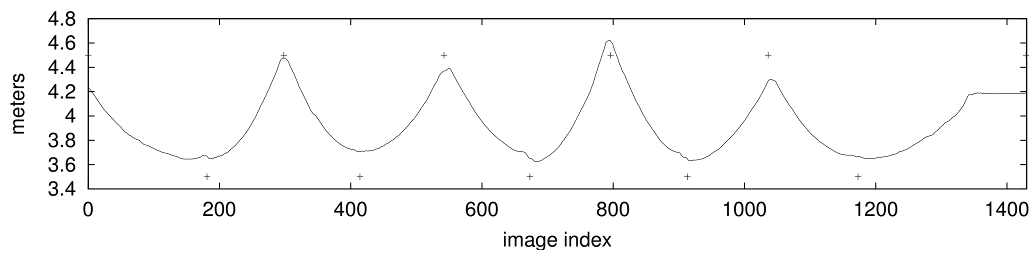


Figure 5.13: The z translation estimates for perspective crane experiment generated by the recursive image-and-inertial method are shown by the line. As in Figure 5.12, the known points are shown as squares.

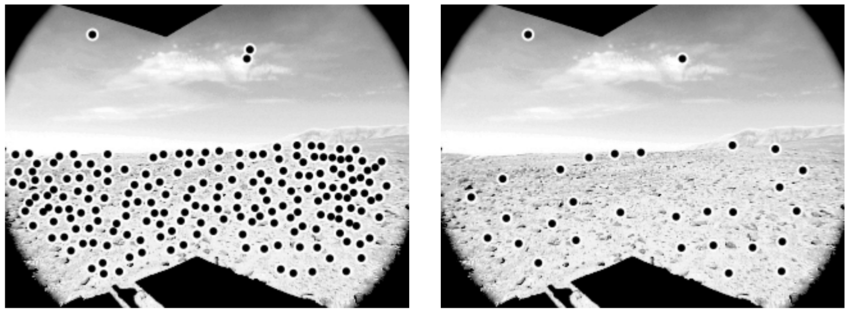
5.6.1 Camera configuration

The sensor rig consisted of an IEEE 1394 camera coupled with a 3.5 mm lens, the gyro, and the accelerometer.

5.6.2 Observations

The sensor rig was mounted horizontally about 1 meter from the ground at the rear of the rover, and rotated forty-five degrees about the vertical axis to look out to the rear and right of the rover. The rover executed three large, roughly overlapping loops during a 15 minutes and 36 second traverse, covering a total distance of roughly 230 meters. During this time, a sequence of 1401 images was acquired at approximately 1.5 Hz, and 112917 inertial readings were acquired at an average frequency of approximately 121 Hz.

The custom tracker mentioned in subsection 2.5 was used to generate two point feature datasets from this image sequence. In the first, 16007 point features were tracked through the sequence, with each image containing an average of 201.5 points, and each point appearing in an average of 17.5 images. That is, each point appears in an average of 1.3% of the image sequence. The density of these features in one image is shown in Figure 5.14(a), and a few features from this dataset are shown over time in Figure 5.15. In the second, a subset of the first point feature dataset containing 5611 point features was considered, where each image contained an average of 34.9 points and each tracked point appears in an average of 8.7 images. This is 0.62% of the image sequence. The relative density of these features in one image is shown in Figure 5.14(b). Some point feature tracks in these datasets exhibit drift, with a maximum cumulative drift on the order of 5 pixels, particularly for features in the distance that can be tracked through many images. However, it appears that there are no gross tracking errors in this dataset of the kind that Lucas-Kanade



(a) (b)

Figure 5.14: An image from the perspective rover sequence with features from the 201.5 features per image dataset (a) and features from the 34.9 features per image dataset (b) overlaid.

often produces.

5.6.3 Recursive image-only estimates

The recursive image-only algorithm described in subsection 4.4.7 was used to generate one motion estimate from each of the two feature datasets. The x, y translation estimates for the 201.5 features per image dataset and for the 34.9 features per image dataset are shown in Figures 5.16(a) and 5.16(b), respectively. The x, y translation estimates from the rover's odometry and from the rover's GPS receiver are shown in Figures 5.16(c) and 5.16(d), respectively. In each diagram a 1 meter wide "x" marks the start position of the rover. The easiest way to understand the rover's qualitative motion is to first examine the estimates in 5.16(a) and 5.16(b), which are smooth everywhere, and then to examine the odometry and GPS estimates in 5.16(c) and 5.16(d), which contain local errors.

Each of these four estimates contains errors of different kinds. The estimates generated by the algorithm, shown in Figures 5.16(a) and 5.16(b), are locally correct everywhere, but exhibit slow drift over time. For the estimate generated using 201.5 features per image, this drift is primarily in x, y translation, while the estimate generated using 34.9 features per image contains drift in both x, y translation and rotation about z. In each case, the drift components in z translation and rotation about x and y, which are not indicated in the figure, are quite small. The estimates from GPS exhibit serious errors, which were common in the GPS data Hyperion acquired and are due to changes in GPS satellite acquisition during the traverse.

The estimates from odometry contain some local errors, visible as small juts on the or-

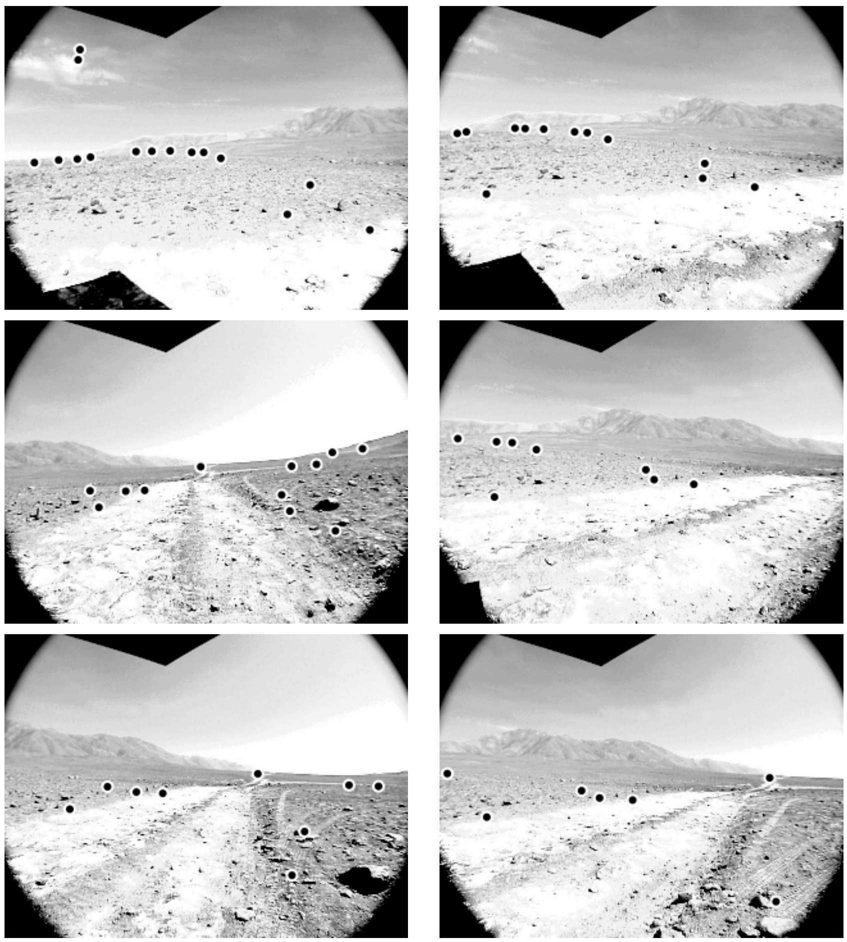


Figure 5.15: Images 150, 160, ..., 200 from the perspective rover experiment are shown clockwise from the upper left, with tracking data from the 201.5 features per image dataset. For clarity, only a number of those tracked features that were visible in a relatively long subsequence of the images are shown.

der of 1 meter in Figure 5.16. These correspond to times in the image sequence where the rover's wheels became caught on a large rock in the rover's path. However, the estimates from odometry are, globally, the best of the four. For instance, the odometry estimate indicates that at the end of the second loop, the rover comes very close to the location it visited at the end of the first loop, as shown in 5.16(c) near (310, 752) where the two loops touch; visual inspection of the image sequence confirms that the rover's position was almost identical during these two times. Similarly, the estimate from odometry indicates that at the end of the third loop, the rover's position is significantly behind the rover's initial position; this also appears to be confirmed by the image sequence. So, the odomoetry estimate has been adopted as the ground truth estimate.

Compared against the odometry estimate, the average and maximum translation errors in the 201.5 features per image estimate, after the scaled rigid transformation, are 1.74 meters and 5.14 meters, respectively. These are approximately 0.8% and 2.2% of the total distance traversed by the rover. The rotation error appears to be negligible in this case. The average and maximum translation errors in the 34.9 features per image estimate are 2.35 meters and 5.27 meters, respectively. These are approximately 1.0% and 2.3% of the total distance traversed by the rover. In this case, an accumulated error in the z rotation is noticeable.

5.6.4 Recursive image-and-inertial algorithm

A collection of estimates from the 34.9 features per image dataset were also computed using the recursive image-and-inertial algorithm. In the collection, the angular velocity and linear acceleration propagation variances were varied from 10^{-4} to 10^{5} and from 10^{-2} to 10^{4}, respectively. As detailed in the paragraphs below and in 5.6.5 below, the contribution of the inertial sensors is limited in this example by the erratic and long-term motion in the sequence.

The rover's sudden accelerations and decelerations are incompatible with linear acceleration propagation variances of 10^{2} or below, and for these tests, the distributions that result from the inertial measurement update steps eventually become incompatible with the subsequent image observations. As a result, most of the image observations are eventually rejected as outliers when these variances are used, and the algorithm fails.

For linear acceleration propagation variances above 10^{2} and angular velocity propagation variances of 10^{4} or below, the resulting estimates are qualitatively similar to the results shown in Figures 5.16(a) and 5.16(b). However, the image-and-inertial estimates show noticeably more drift in rotation in azimuth, and less drift in global scale.

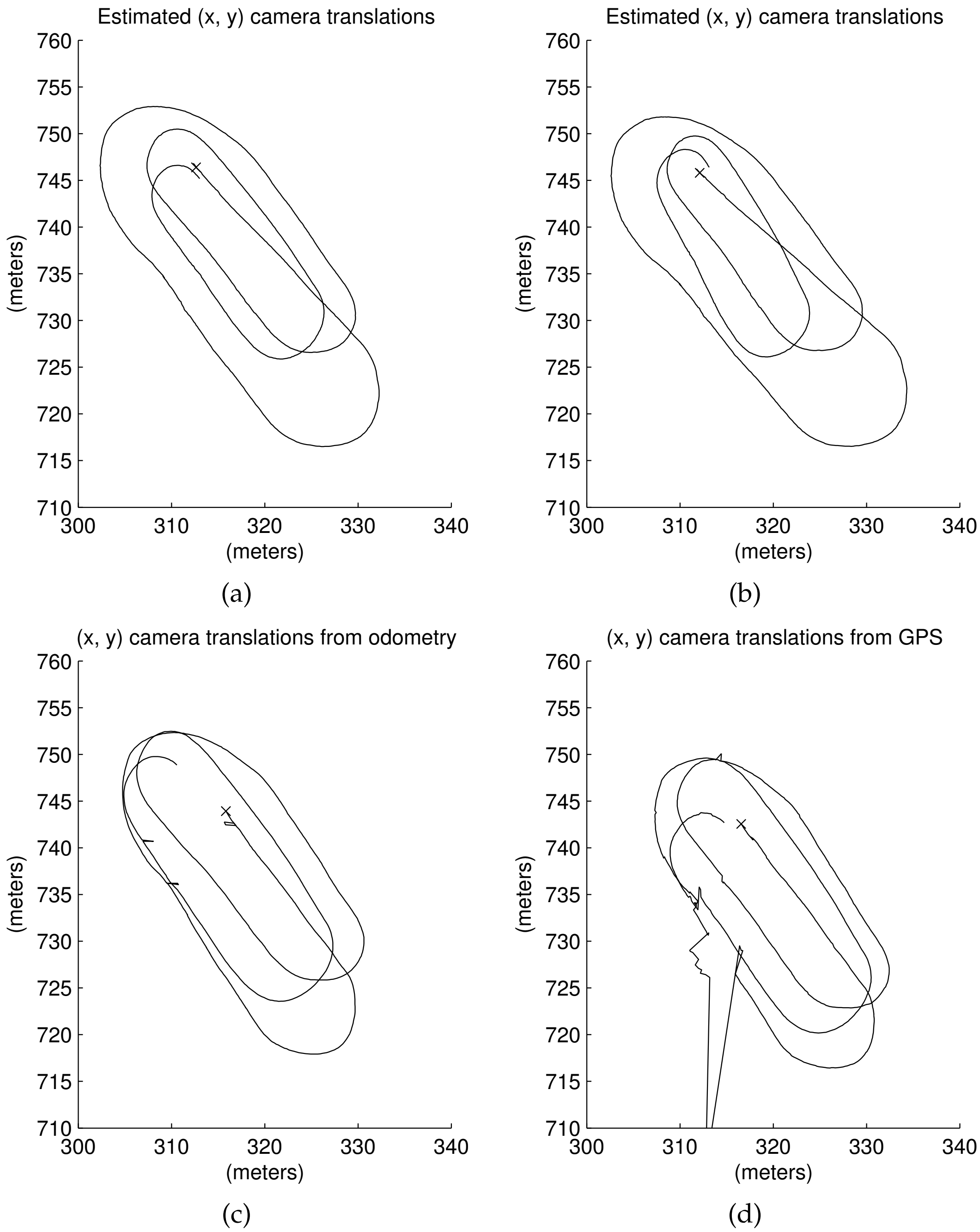


Figure 5.16: The x, y translation estimates the perspective rover dataset, generated using the recursive, image-only algorithm with 201.5 features per image (a); the recursive image-only algorithm with 34.9 features per image (b); the rover's odometry (c); and GPS (d).

As described in subsection 4.4.8, weakening the assumption of continuity in the angular velocity and linear acceleration necessarily weakens the contribution of the inertial measurements in the recursive image-and-inertial algorithm. This is reflected in this experiment: for the most generous propagation variances, i.e., linear acceleration propagation variances above 10^2 an angular velocity variance of 10^5, respectively, the effect of the inertial measurements becomes negligible, and the estimates are almost identical to the 34.9 features per image, recursive image-only estimate shown in Figure 5.16(b).

5.6.5 Discussion

In this experiment, the image-only motion estimates benefit from dense, high-quality feature tracking data, and are locally correct everywhere as a result. However, due to the small number of images in which each point feature is visible, the motion is not being estimated with respect to any one external point, and so necessarily drifts with time.

Long-term drift in the image-only estimates to be a type of ambiguity, in the sense that the true, globally correct motion and a large space of drifting estimates will all produce small reprojection errors. In fact, given that the image observations and the camera calibration both contain small errors, the estimate that best minimizes the reprojection error will almost never be a globally correct estimate, but will contain some drift. This is in contrast to cases such as that described in Section 5.3, where only sparse, noisy tracking data is available, and where even the local motion cannot be uniquely identified from image measurements alone.

Can the incorporation of measurements from inexpensive inertial sensors remove this ambiguity in the long-term motion, if the image-only estimates are locally correct everywhere? Since the sensors' absolute orientation with respect to gravity can be reliably found by separating the effects of gravity and the other unknowns in the accelerometer readings, inexpensive inertial sensors can eliminate long-term drift in the estimated rotation about the two axes perpendicular to the gravity direction. (In the image-only estimates described in subsection 5.6.3, the drift in these two components of the rotation was not significant when compared to the drift in the rotation about the gravity direction and in the x, y translation, so orientation information from the accelerometer was not needed.) However, inexpensive inertial sensors are unlikely to resolve long-term drift in the translation or in the rotation about the gravity direction, because the rate of drift in the inertial sensor estimates is likely to be much faster than that of the image-only estimates.

On the other hand, can the incorporation of measurements from inexpensive inertial sensors improve long-term motion estimates if the image-only estimates are locally ambiguous at some points? Of course, in this case the inertial sensors can improve the long-term motion by resolving these ambiguities, which may turn a qualitatively wrong estimate into an accurate estimate. However, estimates will drift less slowly if the image-only estimates can be made locally correct (e.g., by obtaining dense feature tracking data), than if the image-only estimates are locally ambiguous (e.g., generated using sparse image data) and inertial measurements are incorporated to resolve the local ambiguities.

Chapter 6

Robust feature tracking for motion estimation

6.1 Overview

As the experiments in Chapter 5 show, motion estimation from sparse image features can be sensitive to even small errors in the input projections, even when the motion is estimated using an optimal algorithm. As discussed in subsection 5.3.4, these errors are due to both:

- feature tracking errors
- errors that result from normalizing image measurements with uncertain camera intrinsics

Feature tracking errors might be slowing accumulating errors that result from tracking a point over a long image sequence, or more sudden errors between adjacent images that result from occlusions, lighting changes, specularities, or other difficulties common in image tracking. As a result, the motion estimates produced from the projections can:

- contain local errors
- be locally correct everywhere but contain long-term drift

Local errors are immediate problems because significant errors between adjacent camera position estimates produce large errors in the estimated positions of points visible from those camera positions; in turn, these produce unrecoverable gross errors in the next few camera position estimates. But, long-term drift is a more fundamental problem, since no level of accuracy in the tracking and camera intrinsics can eliminate drift completely. So, image-based motion estimation places high demands on both image feature tracking and camera calibration.

This chapter describes a new tracking algorithm designed to reduce both local errors and long-term drift in image-based motion estimation. The algorithm exploits several observations:

- SIFT keypoints[36], described in Section 2.6, provide a convenient way to estimate the epipolar geometry between images with no problematic assumptions on the camera motion.

- Features do not move arbitrarily between images, as assumed when tracking features individually in two dimensions, but are constrained by the rigid scene assumption. So, each feature should be tracked in one dimension along its epipolar line in its new image.

- Motion estimation benefits from a wide field of view, but the field of view is effectively reduced if features are not tracked throughout the entire image. So, features should be extracted to achieve a wide effective field of view.

- Poorly textured features can often be correctly tracked when constrained to one dimension. So, texture should be a secondary criterion for extraction rather than the primary criterion.

- For image-based motion estimation, features that become clustered during tracking (e.g., by moving towards a common vanishing point) are redundant and should be pruned. Since the oldest feature in an image neighborhood is likely to provide the most information for camera motion estimation, younger features should be pruned away before older features.

- A feature whose appearance did not change between images may be incorrectly tracked (e.g., a feature that jumps between similar areas on a repetitive texture) and a feature whose appearance changed drastically between two images may be tracked precisely, especially when constrained to the epipolar line. So, changes in image appearance should be rejected as a criterion for detecting mistracking.

The resulting algorithm can track with high robustness and accuracy in many sequences for which the previous go-to tracker, Lucas-Kanade, fails. In addition, the resulting distribution of features is more appropriate for motion estimation. As a result of these improvements in feature tracking, local errors in six degree of freedom motion estimates are often completely eliminated.

The remaining sections of this chapter are organized as follows. Lucas-Kanade, which has long been the go-to tracker for shape-from-motion, and the problems in using it for real sequences are reviewed in Section 6.2 and 6.3, respectively. Because the new tracker estimates and uses the epipolar geometry between adjacent images in the sequence, Section 6.4 reviews the epipolar geometry between images and the fundamental matrix that describes it. The design of the new tracker is described in detail in Section 6.5, and experimental results are given in Sections 6.6, 6.7, and 6.8 below. The new algorithm's limitations and possible further improvements are described in 6.9.

As mentioned at the beginning of this section and in Section 2.9, motion estimation accuracy is limited by camera intrinsics errors as well as tracking errors. This work has not investigated improved models for conventional cameras, but better conventional camera models are a promising approach for improving both motion estimation and feature tracking.

6.2 Lucas-Kanade

Lucas-Kanade[37] has long been the preferred sparse feature tracker for shape-from-motion. In brief, Lucas-Kanade finds a template's match location in a new image by using second-order iterative optimization to minimize the sum-of-squared-differences (SSD) between the template and image intensities. Typically, the template's location is taken to be the (x, y) translation, but the SSD is sometimes minimized with respect to an affine or other more general transformation of the template.

To track features in an image sequence taken from a moving camera, Lucas-Kanade is normally used with three heuristics:

- Features that maximize the conditioning of the system solved by Lucas-Kanade on each iteration are extracted for tracking. This approach results in features near high image texture, and rejects features in textureless areas or areas where the texture is one-dimensional.

- A feature is marked as mistracked and discarded if the Lucas-Kanade iterations fail to converge, or if the SSD error at the minimum is larger than a predefined threshold. In the second case, the feature is marked as mistracked because it has changed appearance.

- Large motions are accommodated using a translational pyramid. In this approach, the image size is reduced multiple times until the expected maximum feature motion, in pixels, is small enough to be tracked using Lucas-Kanade starting from the feature's position in the previous image. The resulting estimate for the feature's new position is then scaled up and used as the initial position estimate for Lucas-Kanade in the next finer resolution image of the pyramid. This process is continued until the motion estimate has been propagated back to the original resolution image at the bottom of the pyramid.

When used with these heuristics, Lucas-Kanade has some compelling advantages. Often the minimization converges after just a few iterations, so that the number of SSD evaluations is small, making the method faster than correlation search over a large image area. The method tracks to subpixel resolution, and one-tenth of a pixel between adjacent images in a sequence is often cited as a typical accuracy. Furthermore, Lucas-Kanade is essentially general minimization applied to the SSD error, so it can be easily generalized to expand the parameterization of the feature's location in the new image, or to incorporate priors on the feature's motion.

As an example, the begin and end images of a 20 image subsequence of the classic "hotel" sequence are shown in Figure 6.1, along with a few of the 100 features tracked across the subsequence using Lucas-Kanade. To get a measure of the tracking accuracy for this sequence, the following steps have been used. Factorization followed by bundle adjustment have been applied to the tracked points, with the bundle adjustment observation covariances for the projections in the first image chosen several orders of magnitude smaller than those for subsequent images, i.e., the projections in the first image are taken to be exact. The reprojection errors for each point in the final image are then taken to be the accumulated tracking error over the subsequence. This error is an approximation to the tracking error in that the uncertain camera intrinsics also contribute some error, but this error is likely to be small in this sequence because the sequence contains negligible radial distortion.

Lucas-Kanade is highly accurate for this sequence, as shown in Figure 6.2, which gives the distribution of tracking errors for the 100 tracked features. The tracking error for all features is less than 1.6 pixels, with all features except 4 less than 1 pixel. A significant number of features, 15, have accumulated tracking errors across the sequence of less than 0.1 pixels. This is an order of magnitude better than the Lucas-Kanade folk accuracy of 0.1 pixels between adjacent images in a sequence.

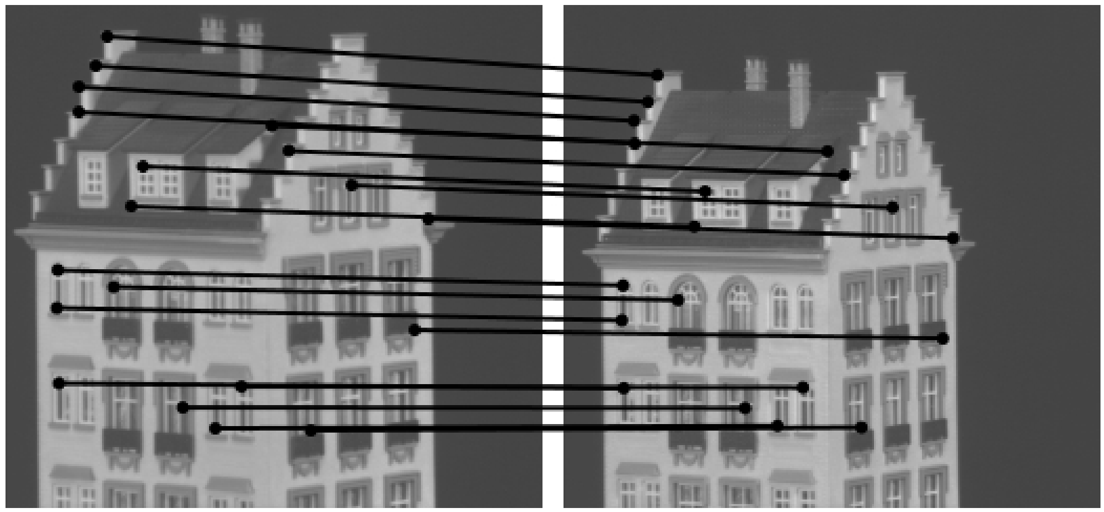


Figure 6.1: Features tracked across 20 images of the classic "hotel" sequence.

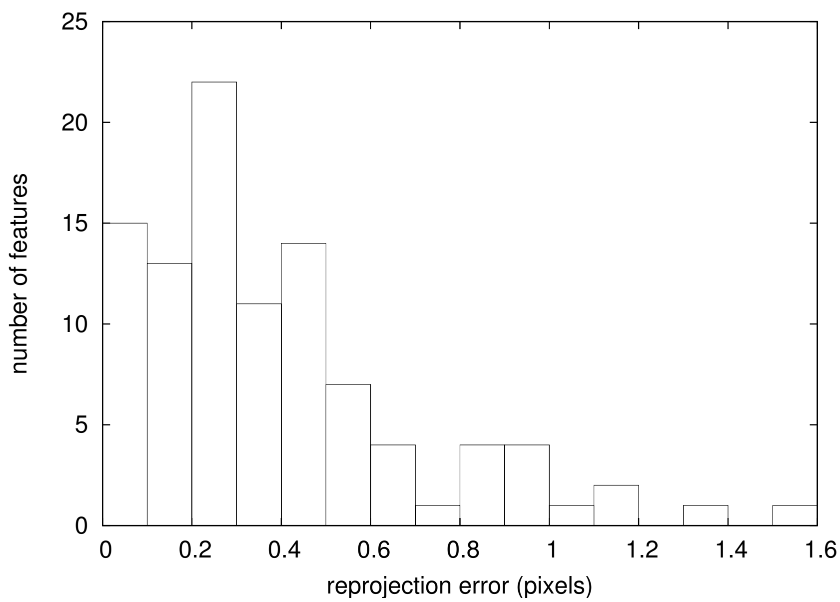


Figure 6.2: The distribution of Lucas-Kanade tracking errors for the "hotel" subsequence.

6.3 Lucas-Kanade and real sequences

The hotel sequence described in the previous section is benign in many ways. The features' motion between images is nearly uniform in (x, y), and therefore well modeled by Lucas-Kanade's translational pyramid, but also small enough that the pyramid is not necessary. While the image texture is repetitive, the frequency of the repetition is low relative to the features' small motion. In addition, the features do not undergo any deformation, occlusion, or change in overall intensity. So, drift is minimal and Lucas-Kanade's image-based heuristic for detecting mistracked points works well for this sequence.

For many sequences taken from a moving camera, however, Lucas-Kanade produces tracking that is unusable for motion estimation. Lucas-Kanade's poor performance results from:

1. *Unconstrained tracking.* The locations of features tracked without constraints are unduly sensitive to changes in appearance, occlusion, and low texture.

2. *A poor heuristic for handling large motions.* Many overall image motions are not well modeled by the translational pyramid. For instance, camera rotations about and camera translations along the optical axis produce image motions that average to zero (x, y) translational motion over the image, but are actually zero at almost no specific locations in the image. Furthermore, while the translational pyramid allows the tracking of some motions that are too large to be tracked with bare Lucas-Kanade, the translational pyramid sets an a priori limit on the size of motions that can be tracked.

3. *A poor heuristic for feature extraction.* As mentioned in Section 6.1, exploiting the camera's full field of view for motion estimation requires that features be extracted and tracked correctly throughout the entire image, even if some areas of the image have only poor texture.

4. *A poor heuristic for discarding features.* As mentioned in Section 6.1, changes in feature appearance, or lack thereof, are poor indications of mistracking.

Some image sequences that are problematic for Lucas-Kanade have already been shown in Chapter 5. The image sequence from the perspective arm sequence, shown in Figure 5.2, is one example. In this sequence, the large rotation about the optical axis produces large motions in the periphery of the image that are not captured by Lucas-Kanade's translation pyramid. The unmodeled motion is especially damaging for features on the light grates and on the air vent, which are repetitive textures, and as a result, features jump between locations on the grates and air vent. However, tracking is also poor on the styrofoam ceiling tiles, which have poor texture, and on the metal slats that support them, which are one-dimensional textures.

The perspective crane image sequence shown in Figure 5.11 is a second example. In this sequence, Lucas-Kanade's translational pyramid model is appropriate. However, tracking in this sequence suffers in this sequence from repetitive texture, and gradual occlusion of features, and specularities.

A third example is the perspective rover image sequence shown in Figures 5.14 and 5.15. In this example the translational pyramid again fails, since for much of the sequence all of the terrain points move towards a single vanishing point on the horizon. This motion also produces clumps of points in a single area of the image near the horizon. Lucas-Kanade's heuristics for eliminating and extracting points also fails, producing a redundant set of points and leaving the rest of the image unpopulated.

6.4 Background: epipolar geometry and the fundamental matrix

As described in Section 6.5 below, the new tracker estimates the epipolar geometry between adjacent images in the sequence and uses it to constrain the tracking between adjacent images. In this subsection we briefly review the epipolar geometry between two images and the fundamental matrix that describes it.

The epipolar geometry between two images is illustrated in Figure 6.3. Suppose that a point X is visible in the first image and that the camera's motion between the times of the first and second images are known, but that the depth of X is unknown. Then an epipolar line l' in the second image that must contain the projection x' of X can be found by backprojecting a ray from the first camera center c through the projection x of X in the first image, and then reprojecting the ray into the second image.

Since all of the rays backprojected from the first image originate from the camera center at the time of the first image, and the epipolar lines are reprojections of these rays, all of the epipolar lines in the second image must intersect at the projection of the first image's camera center in the second image. This point is called the epipole and is marked as e' in Figure 6.3.

The essential matrix E relates corresponding normalized projections $x = [x_0\, x_1]$ and $x' = [x'_0\, x'_1]$ in the two images through the equation:

$$[x_0\, x_1\, 1]E[x'_0\, x'_1\, 1]^{\mathrm{T}} = 0 \tag{6.1}$$

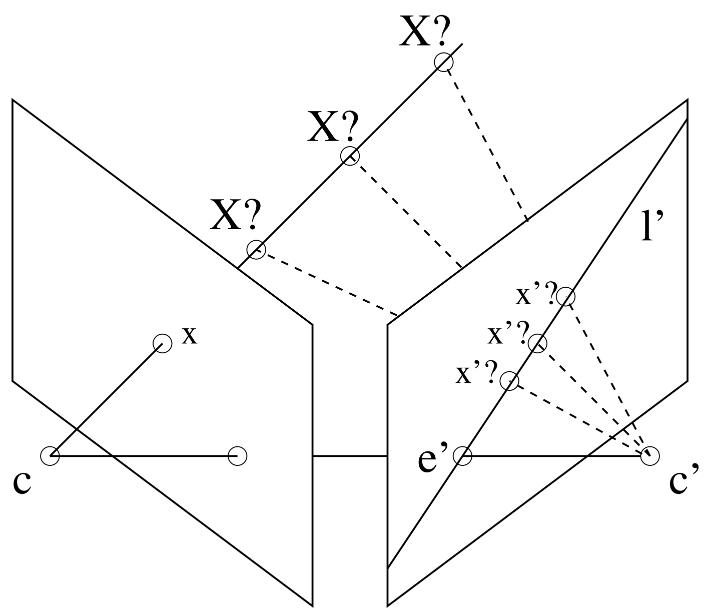


Figure 6.3: The epipolar geometry relating two images, adapted from Hartley and Zisserman[26].

In projective geometry, a two-dimensional line l' is represented as a three-vector $\hat{l}$, where $\hat{l}x' = 0$ for points x' on l'. So, for the projection x of X in the first image, (6.1) gives the epipolar line l' containing x' as $\hat{l} = [x_0 \, x_1 \, 1]E$.

If the transformation R, t between the two image positions is known, then the essential matrix is:

$$E = [t]_\times R \tag{6.2}$$

where $[t]_\times$ is the skew-symmetric matrix:

$$[t]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \tag{6.3}$$

As mentioned above, the essential matrix E relates normalized projections in the two images. The fundamental matrix F is a generalization of the essential matrix that can relate projections in unnormalized image coordinates. If the relative position between the two images and the intrinsics are both known, then the fundamental matrix is:

$$F = [t]_\times K' R K^{-1} \tag{6.4}$$

where

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1.0 \end{bmatrix} \tag{6.5}$$

F can be then be used to generate a projective description of the line in the second image corresponding to a point x in the first image.

The fundamental matrix cannot be used to relate corresponding points in images with radial distortion, since in this case the reprojection of the ray containing x and X in the second image is not a straight line.

6.5 Tracker design and implementation

6.5.1 Overview

The new tracker is designed specifically for image-based motion estimation applications, and resolves problems with Lucas-Kanade by exploiting the rigid scene constraint and eliminating Lucas-Kanade's heuristics. The main steps of the algorithm are epipolar geometry estimation, epipolar search, geometric mistracking detection, and feature thinning and extraction, as shown in Figure 6.4 and detailed in subsections 6.5.2- 6.5.5 below.

This section includes some images from the perspective rover sequence, introduced in Chapter 5 for evaluating motion estimation from image and inertial measurements, to illustrate the tracker components. Detailed tracking results from the perspective rover sequence and other test sequences are given in Sections 6.6, 6.7, and 6.8 below.

6.5.2 Epipolar geometry estimation

As shown in Figure 6.4, when a new image arrives, the tracker first estimates the epipolar geometry between the current image and previous image, so that the features' image motion can be constrained. For images from a video sequence, F can be estimated as follows.

1. Extract SIFT keypoints in the current image. As Section 2.6 describes, SIFT keypoints are image interest points that can be extracted with subpixel accuracy and high repeatability despite changes in image scale and rotation. Each SIFT keypoint has an associated feature vector, computed from the image intensities in the neighborhood of the keypoint, that allows it to be matched with high accuracy to the corresponding keypoint extracted in other images, without prior knowledge of the relative imaging geometry between the images. SIFT features are described in detail in Lowe[36].

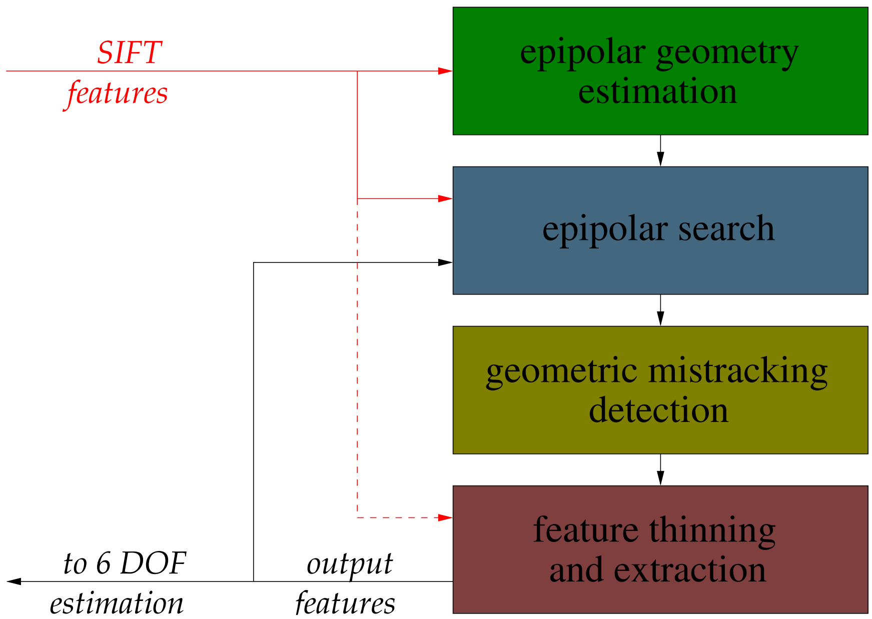


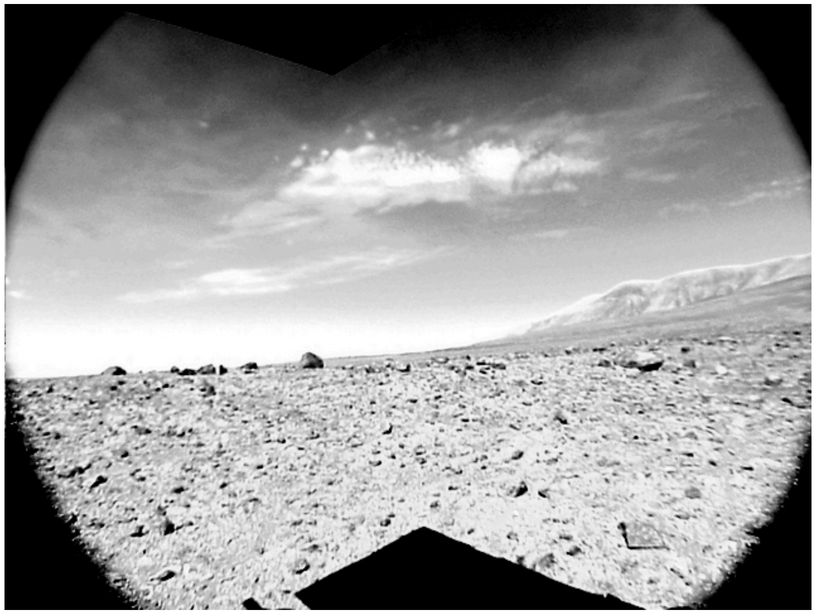
Figure 6.4: An overview of the tracker design.

The SIFT features extracted for one of the example rover images are shown in Figure 6.5.

2. Mask the keypoints. If a mask is available specifying areas of the image that are known to be on objects moving independently of the rigid scene, keypoints in those areas are eliminated. The current implementation accommodates problems where:

- There is no mask, because everything visible in the images is part of the rigid scene.
- There is a mask, and it is the same for every image. This is the case when, for example, the camera is rigidly attached to a vehicle that is partially visible in the images.
- There is a mask, and it changes from image to image.

In the rover sequence used as an example in this section, a filter ring attached to the camera lens, the rover's solar panel, and the rover's shadow are all visible in the images. While the filter ring and the rover's solar panel are rigid with respect to the camera and always visible in the same part of the images, the position of the rover's shadow in the images changes with the rover's orientation. So, the mask must change on every image. Generally, the masks have to be supplied by some external routine designed for the particular



(a)

(b)

Figure 6.5: An input image from the rover sequence (a) and the SIFT keypoints extracted on the image in step 1 (b). In (b), the black keypoints and white keypoints are those allowed and rejected by the keypoint masking in step 2.

application, and the masks used with the rover sequence were generated using an ad hoc combination of thresholding, morphology, and connected components analysis.

3. Match keypoints using their feature vectors. Each masked keypoint a in the current image is compared against each masked keypoint b from the previous image. Keypoint a is matched to b if the Euclidean distance between the feature vectors for a and b is less than that between a and any other keypoint in the previous image, and if the distance is less than some prespecified threshold.

4. Limit the number of keypoint matches. If the number of keypoint matches from step 3 is greater than some fixed number n, limit the number of matches by selecting a random subset of n matches. In the experiments in this work, n is 100 or 200. Figure 6.6 shows 100 randomly selected SIFT matches between the first perspective rover image and the second image.

5. Find consistent matches. Use RANSAC and two-image bundle adjustment to find a large set of consistent keypoint matches between the previous and current image. In the example in Figure 6.6, all of the matches are consistent.

6. Estimate the fundamental matrix. A final bundle adjustment is performed using all of the consistent matches to estimate the six degree of freedom camera motion between the previous and current image times. Then, using the estimated camera motion, the known camera intrinsics, and Equation (6.4), the fundamental matrix relating projections in the current and previous images is computed.

As mentioned in Section 6.4 above, the fundamental matrix cannot relate projections in images that contain radial or tangential distortion. So, the intrinsics used to compute the fundamental matrix in this step are those that result from zeroing the radial and tangential distortion terms in the camera's actual intrinsics. As a result, the computed fundamental matrix relates projections in images that have been corrected to remove the radial and tangential correction, and tracking along epipolar lines is performed in the corrected images, as described in subsection 6.5.3 below. Figure 6.7 below shows the effects of this correction on two images.

Figure 6.8 shows several epipolar lines computed for the example pair using the estimated fundamental matrix and the projections of several points in the first image. In each case, the corresponding projection in the second image lies on the epipolar line.

As mentioned in Section 6.4 above, the epipole in the second image is the projection of the first image's camera center in the second image, and occurs where the epipolar lines in the second image intersect. Depending on the motion, the epipolar may or may not lie

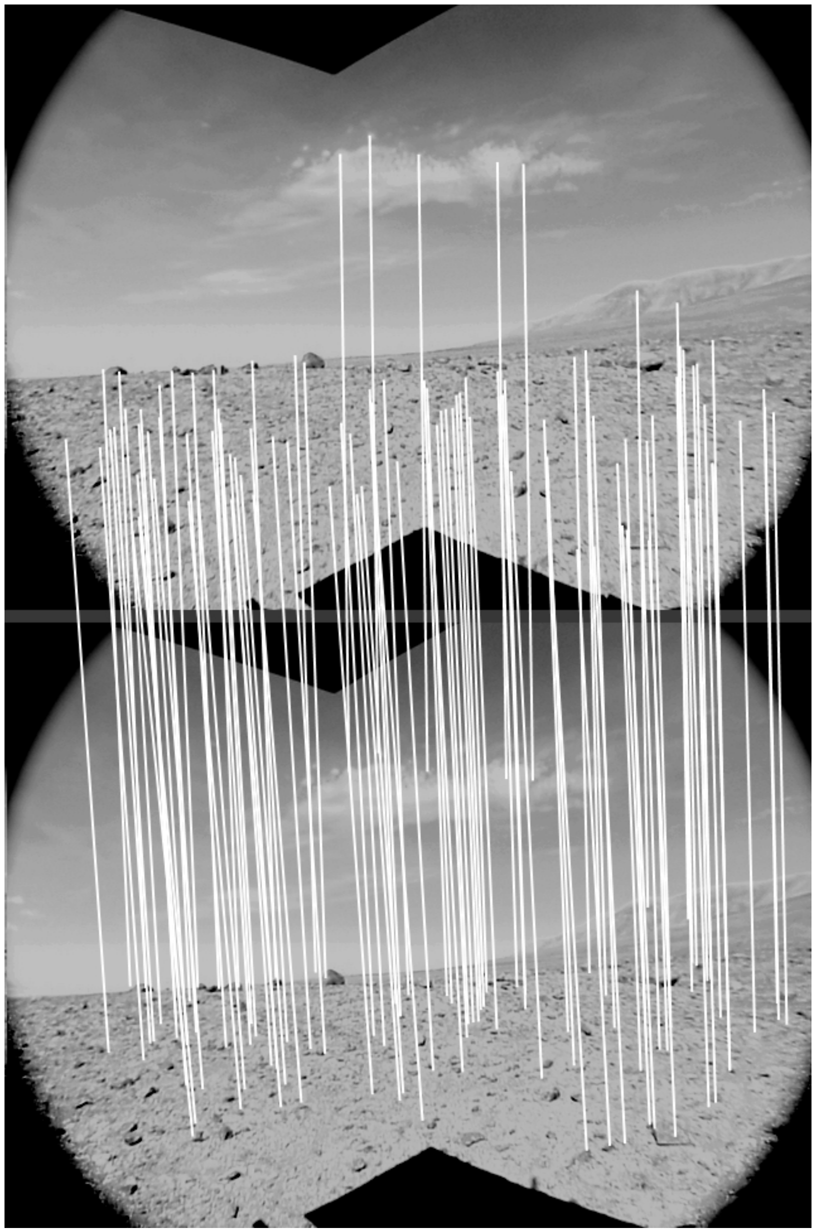


Figure 6.6: 100 randomly subsampled matches between SIFT features in the first perspective rover image (top) and the second image (bottom). In this example, all 100 of the matches are consistent.

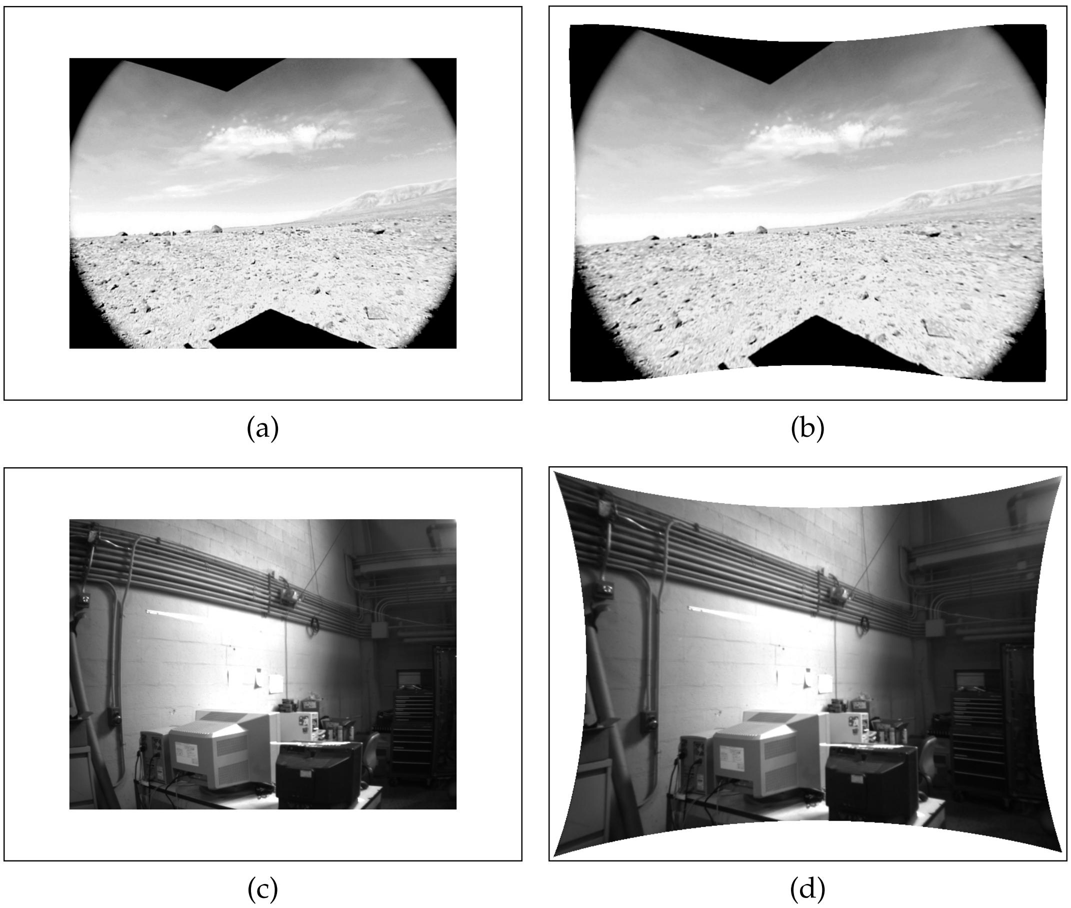
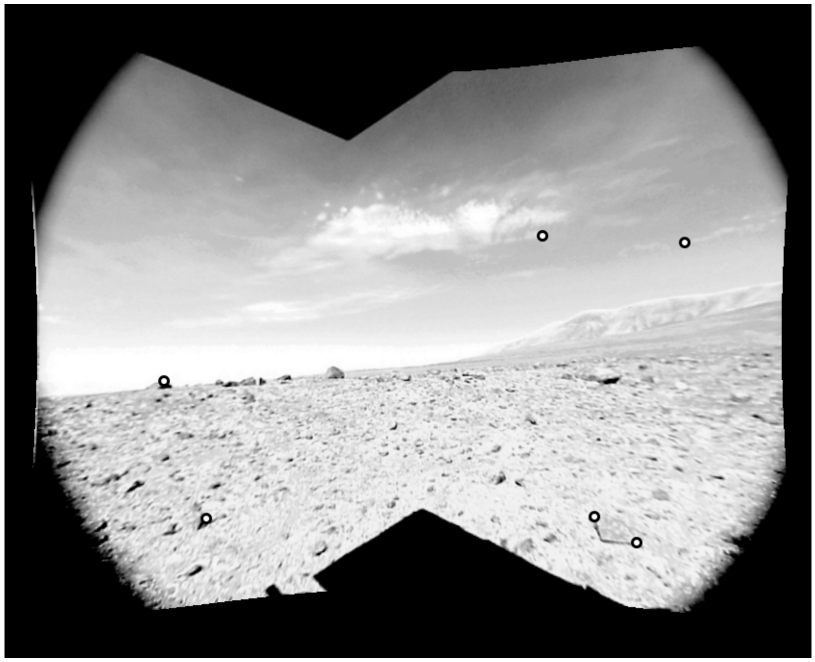
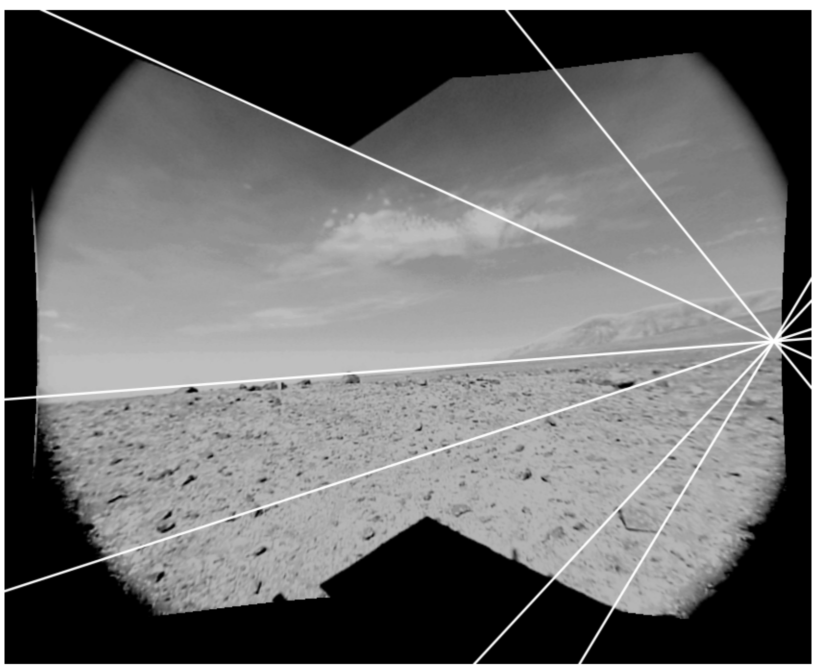


Figure 6.7: As described in subsection 6.5.2, the fundamental matrix cannot relate images that contain radial or tangential distortion, so the images must be corrected for radial and tangential distortion before tracking along epipolar lines. The images in (a) and (b) are uncorrected and corrected images from the example rover sequence. The images in (c) and (d) are the uncorrected and corrected images from a sequence that shows the effect more clearly. For example, the curved pipes high on the wall in (c) are straightened in (d).



(a)



(b)

Figure 6.8: A few example feature locations from the first corrected rover image are shown in (a), and the corresponding epipolar lines are shown in the second corrected rover image in (b). For each feature location in (a), the corresponding location in (b) is on the epipolar line. The epipole in the second image, which is at the intersection of the epipolar lines, is in the image in this example.

within the image. For the motion estimated between the two example images, the epipole is in the image, as shown in the figure.

As mentioned in subsection 6.1, using SIFT to estimate the epipolar geometry is advantageous in that SIFT makes no problematic assumptions on the image motion. For instance, SIFT does not place any a priori limit on the size of the motion, or assume that the motion is continuous or translational.

6.5.3 Epipolar search

Once the fundamental matrix relating the two undistorted images has been recovered, the locations of features in the current image are found by a one-dimensional search along their epipolar lines, as described in this subsection. This is the second box in Figure 6.4.

The following steps are used to perform the one-dimensional search.

1. Get an initial position. Get an initial position in the new image for each tracked point by taking an average of the motion of the inlier SIFT keypoints from the previous step. The terms in the average are weighted by the inverse distance of the SIFT keypoint's position in the first image from the tracked point's position in the first image.

Points whose resulting initial positions are outside of the image, or whose template from the previous image would overlap with the invalid mask if placed at the initial estimate, are marked as invalid and removed from the tracking process. This prevents points from latching onto objects in the image that are not part of the rigid scene.

2. Perform a discrete search. For each surviving feature, project the features's initial position found in step 1 onto the features's epipolar line in the current image. Then, search discrete locations along the epipolar line on each side of this projected location for the sum-of-squared-differences minimum between the feature's intensity template from the previous image and the intensities in the current image.

The existing implementation uses 60 locations on each side of the projected location in 1 pixel increments, and a 41×41 pixel feature template from the previous image, for the discrete search. The initial estimates generated from SIFT matches in step 1 above may or may not be accurate depending on the amount of parallax in the scene and the distance from the tracked feature to the nearest SIFT keypoints, so the resulting 120 pixel search range is not overly conservative. The large 41×41 template sizes are chosen to eliminate ambiguity in the matching.

As in the previous step, features whose resulting positions are outside of the image, or whose template from the previous image would overlap with the invalid mask if placed at the initial estimate, are marked as invalid and removed from the tracking process.

3. Perform a subpixel search. To get the feature's location to subpixel accuracy, perform a one-dimensional Lucas-Kanade search along the epipolar line, starting from the discrete minimum found in the previous step.

In the current implementation, a 21×21 pixel template is used for the one-dimensional Lucas-Kanade search.

6.5.4 Geometric mistracking detection

As shown in Figure 6.4, the third component of the tracker is geometric mistracking detection.

Once the features have been tracked into the current image using one-dimensional search, RANSAC and bundle adjustment are used to identify the largest set of features that appear in the most recent three images, and whose tracked locations in those images are consistent with some choice of six degree of freedom camera positions for those images and three-dimensional point positions for the tracked features. Features that appeared in the three most recent images and are not consistent with this set of feature locations are identified as mistracked, and removed from the tracking process starting with the current image.

6.5.5 Feature thinning and extraction

As shown in Figure 6.4, the final step of the tracker is feature thinning and extraction.

1. Feature thinning. As features are tracked through the image sequence, their relative positions in the image will change, and if new features are extracted in newly visible parts of the scene as the sequence progresses, an arbitrarily large number of features may accumulate in one part of the image. For instance, if the camera is moving backward along its optical axis, tracked points will group together at the center of the image. Tracking all of these grouped points is impractical, and for the purposes of camera motion estimation, tracking all of these points is not necessary, since the information these points provide is nearly redundant.

So, features that have survived geometric mistracking detection are thinned by eliminating features whose locations in the current image are within some threshold distance of the location of an older surviving feature. Removing younger features is preferred because older features are likely to be more valuable in estimating the camera position relative to the camera's start position.

2. Feature extraction. As the image sequence progresses, new areas of the scene will become visible, and already visible parts of the scene may become featureless as mistracked features are detected and eliminated. So, feature extraction is performed in each image by choosing features in areas of the current image that are featureless after thinning.

As mentioned in Section 6.1, feature extraction is based on two observations:

- Large fields of view are beneficial for estimating six degree of freedom camera motion, but to exploit the large field of view it is necessary to track features throughout the entire view.

- Constraining feature tracking to the feature's epipolar lines in the current image makes tracking less sensitive to poor, repetitive, or one-dimensional texture than unconstrained tracking.

So, the feature extraction works by choosing features that are at least some distance from any existing feature, and as a secondary criterion, have some minimal texture.

In the current implementation, the SIFT keypoint locations are used as the candidate feature locations, because SIFT keypoints tend to be extracted wherever there is some minimal texture in the images, as shown in Figure 6.5(b). However, this is not a fundamental choice, and any features that satisfy these two criteria would be acceptable choices. This is indicated by the dashed line in Figure 6.4.

To prevent newly extracted points from being thinned in the very next image, the threshold distance for thinning should be smaller than that for extraction. 15 and 25 pixels are typical thresholds for thinning and extraction, respectively.

6.6 Results Overview

The tracker's behavior has been evaluated on five image sequences, which differ in:

- camera configuration

- amount of translational and rotational camera motion

- scene planarity or nonplanarity, and range of scene depths

- image conditions, such as amount and type of texture, occlusions, and specularities

The five sequences are:

- **Rover.** This sequence includes the first 200 images from the perspective rover sequence examined in the previous chapter. The images are 640 $\times$ 480, taken from an IEEE 1394 camera with a wide field of view lens, mounted on a manually controlled rover.

  The camera motion for this sequence is roughly planar translation in x, y and rotation about the camera's y (image up) axis. The scene includes roughly planar terrain in the foreground, with mountains and the sky at infinity, and contains a mix of well textured and weakly textured areas.

- **Highbay.** This sequence includes 200 640 $\times$ 480 images taken from an IEEE 1394 camera with a wide field of view lens, mounted on a manually controlled cart.

  Like the rover sequence, the camera motion for this sequence is planar translation in x, y and rotation about the camera's y (image up) axis. The scene is highly non-planar and well textured in most areas. The resulting sequence includes occlusions.

- **Helicopter.** This sequence includes 200 640 $\times$ 480 images taken from an IEEE 1394 camera with a wide field of view lens, mounted on a remote control helicopter.

  The camera motion in this sequence contains translation in x and y, a lesser translation in z, and rotation about the camera's optical axis. The scene is planar, and includes a mix of well textured and weakly textured areas. The images contain severe compression artifacts.

- **Crane.** This sequence contains 200 images selected from the perspective crane sequence examined in Chapter 5. The images are 640 $\times$ 240, taken from a CCD camera with a narrow field of view lens, mounted on a computer controlled crane.

  The camera motion includes significant translational motion in x, y, and z, but only very minor rotational motion. The scene is highly nonplanar, well textured everywhere, and contains repetitive texture, one-dimensional texture, and specularities. The resulting image sequence contains many occlusions.

- **Arm.** This sequence is the perspective arm sequence examined in Chapter 5. The sequence includes 152 640 $\times$ 240 images taken from a CCD camera with a narrow field of view lens, mounted on a computer controlled arm.

  The camera motion includes significant translational motion in x, y, and z and significant rotational motion about the camera's z (optical) axis. The scene is planar, and contains repetitive, weak, and one-dimensional texture.

	number of images	image resolution	camera type	planar scene	nonplanar scene	translational motion	rotational motion	well textured	poorly textured	repetitively textured	one-dimensional textures	specularities	occlusions	severe compression artifacts
rover	200	full	1394	✓	✓	x, y	y	✓	✓					
highbay	200	full	1394		✓	x, y	y	✓					✓	
helicopter	200	full	1394	✓		x, y, z	z	✓	✓					✓
crane	200	half	CCD		✓	x, y, z		✓		✓	✓	✓	✓	
arm	152	half	CCD	✓		x, y, z	z		✓	✓	✓	✓		

Table 6.1: The characteristics of the test image sequences. Here, "full" and "half" image resolution are 640 $\times$ 480 pixels and 640 $\times$ 240 pixels, respectively; camera type "1394" refers to IEEE 1394 cameras; x, y, and z translation axes are world (scene) axes; and y and z rotation axes are camera axes (image up and optical axes, respectively).

Table 6.1 summarizes the characteristics of these five sequences.

Sections 6.7, 6.8, and 6.9 below describe the evaluation methodology, the evaluation results, and possible future directions, respectively.

6.7 Evaluation methodology

Section 6.8 below describes the following aspects of the tracker's behavior for each of the five image sequences:

1. Basic statistics, including the number of features tracked; a breakdown describing the features' "fates", i.e., how tracked features were lost; and the distribution describing how many images features were tracked through.

2. An estimate of how many features were grossly mistracked and how many features were tracked correctly up to drift.

3. Estimates of the tracking error and drift rate for those features that were not grossly mistracked.

This section briefly describes how each of these statistics is computed.

6.7.1 Feature fates

For 1, tracked features are subject to four fates. Three of these fates are for lost features, which can be thinned, leave the field of view, or be eliminated because of geometric inconsistency. The fourth fate is survival until the end of the sequence.

6.7.2 Estimating correct feature positions

For 2 and 3, estimates of the correct feature positions are needed. These are found by (1) estimating each feature's three-dimensional position from estimates of the camera position at the time of each image and from the tracked features and (2) reprojecting the three-dimensional position back into the images using the camera position estimates.

Two different approaches to computing the reprojections have been used. In the first, the observation covariance associated with a feature in the first image in which it is visible is set very low, and the observation covariances are set normally for all subsequent images. As a result, the reprojection for the first image is the same as the observed projection, and the reprojection error is distributed across the remaining features, but not necessarily in the increasing "drifting" manner that is likely to be the real errors' behavior. The estimated tracking errors that result from these reprojections serve as an approximate lower bound on the real tracking errors. In the text below, reprojections computed using this method are called "1 covariance" reprojections.

In the second, the observation covariance for the first image is set very low, the covariances for the next nine images slightly higher, and the covariances for all subsequent images are set normally. In this scenario, the reprojection for the first image is the same as the observed projection, reprojection errors for the next nine images are not necessarily "drifting", and the reprojection errors for all subsequent images are likely to exhibit the correct "drifting" behavior. The tracking error estimates computed from these reprojections are likely to be closer to the true tracking error than the 1-covariance reprojections. These reprojections are called the "10 covariances" reprojections in the text below.

A third method, "2 covariances", was also tried. In this method, the first observation covariance is set very low, the second observation covariance was set slightly higher, and all subsequent covariances were set normally. Using these covariances, the first and second reprojection are nearly the same as the first two observed projections, and the three-dimensional point position is essentially triangulated using the first two observations. However, this method uses too little data and often results in poor reprojection estimates.

For the rover, highbay, helicopter, and crane image sequences, the camera position estimates were found by supplying the tracking data to a variable state dimension filter equipped with aggressive outlier detection. In the arm sequence, the camera position estimates are the "$\star$" image-and-inertial batch estimates described in subsection 5.3.3.

6.7.3 Identifying grossly mistracked features

For 2, features whose tracking errors only increase or decrease slowly are identified as drifting features, whereas features whose tracking errors change more than some fixed number of pixels between any two images are identified as grossly mistracked features. Here, the tracking errors are treated as two-dimensional, directional errors rather than scalars. So, for example, the change in error of a feature that is observed three pixels below its correct position in one image and three pixels above its correct position in the second image is 6.0 pixels, rather than 0.0 pixels.

Three different levels of gross mistracking have been explored, corresponding to 3.0, 6.0, and 9.0 pixel thresholds for the change in error. The 3.0 pixel threshold is the most reasonable and, for brevity, detailed results have only been reported below for this threshold. The other thresholds are useful for identifying those features that exhibit the most severe mistracking.

6.7.4 Drift estimates

For 3, tracking errors are summarized for those features that are identified as drifting. Drift rate statistics are also given, which normalize the reprojection error with the number of images since the feature was first visible.

6.8 Evaluation

Table 6.2 gives the basic statistics describing the tracker behavior for each of the five test sequences. The first five entries of the table show how many features were tracked in each sequence (row 1), the fates of those features that were eliminated during tracking (rows 2, 3, and 4), and the number of features that were tracked until the end of the sequence (row 5). Rows 6-9 summarize the distribution describing the number of images the features were tracked through. For instance, 25% of the features tracked in the rover sequence were tracked through 16 images of more, and one of those features was tracked through 167 images.

	rover	highbay	helicopter	crane	arm
total features	2324	7041	8075	3053	1692
thinned features	1498	448	311	346	117
features that left the view	324	2679	3691	1726	1080
inconsistent features	323	3762	3799	863	345
surviving features	179	152	274	179	150
75% of features are visible in images:	≥ 5	≥ 3	≥ 3	≥ 3	≥ 4
50% of features are visible in images:	≥ 9	≥ 3	≥ 4	≥ 7	≥ 12
25% of features are visible in images:	≥ 16	≥ 6	≥ 8	≥ 18	≥ 25
one feature is visible in images:	167	48	43	72	113

Table 6.2: The first five entries of the table show how many features were tracked in each sequence (row 1), the fates of those features that were eliminated during tracking (rows 2, 3, and 4), and the number of features that were tracked until the end of the sequence (row 5). Rows 6-9 summarize the distribution describing the number of images the features were tracked through. For instance, 25% of the features tracked in the rover sequence were tracked through 16 images of more, and one of those features was tracked through 167 images.

Table 6.3 gives the percentages of grossly mistracked points for each of the test sequences. For each test sequence, the percentage of features that were classified as grossly mistracked using the 1 covariance and 10 covariances reprojections are given.

Table 6.4 gives the tracking errors and drift rates for drifting features tracked in the test sequences, on the left and right, respectively. Both the tracking errors and drift rates are given for the 1 covariance and 10 covariance reprojections. The average tracking errors or drift rates are given before the slash, and the maximum tracking error or drift rate is given after the slash.

For many real sequences where Lucas-Kanade fails, this tracker is able to generate ac-

	1 covariance	10 covariances
rover	0.15%	0.20%
highbay	11.05%	13.01%
helicopter	7.22%	7.19%
crane	6.99%	7.14%
arm	10.54%	10.69%

Table 6.3: For each test sequence, the percentage of features that were classified as grossly mistracked using the 1 covariance and 10 covariances reprojections.

	tracking errors (pixels)		drift rate (pixels/image)	
	1 covariance	10 covariances	1 covariance	10 covariances
rover	0.84/36.87	1.25/39.14	0.25/2.68	0.23/2.58
highbay	1.52/18.93	1.63/22.77	0.67/2.99	0.66/2.99
helicopter	1.18/12.26	1.43/18.05	0.62/2.95	0.61/2.95
crane	1.36/15.07	2.22/42.07	0.46/2.98	0.44/2.98
arm	2.45/18.54	3.41/50.04	0.55/2.90	0.53/2.99

Table 6.4: The tracking errors and drift rates for drifting features tracked in the test sequences are shown on the left and right, respectively. Both the tracking errors and drift rates are given for the 1 covariance and 10 covariance reprojections. The average tracking errors or drift rates are given before the slash, and the maximum tracking error or drift rate is given after the slash.

curate tracking over long sequences, without manual correction. In many cases, this makes it possible to estimate motion over long sequences without any operator intervention or tweaking. For example, in Chapter 7 below, this ability and SIFT-based reacquisition for limiting drift in sequences where the camera revisited previously mapped locations, are demonstrated on the full "highbay" sequence of 945 images.

An additional test was performed for each dataset, in which features were tracked forward through a subsequence of the images, and then back through the same subsequence. So, the starting and ending images of the resulting synthetic sequence are the same image, which makes a direct measure of the accumulated tracking error possible.

The results are shown in Table 6.5 below. In each case, the subset of the sequence was chosen so that features were visible in each of a relatively large number of images. In some cases, however, features are never visible in more than a very short subsequence. For example, in the highbay sequence, 14 images was the longest subsequence found in which a reasonable number of features were visible throughout the entire subsequence.

The overall results are excellent, with the average drift rates (i.e., tracking error between adjacent images) varying between 0.040 pixels and 0.15 pixels.

6.9 Future directions

The robustness and accuracy of the tracker described in this chapter can be improved in a number of ways. Highest priority first, these are:

1. Even when tracking is constrained to the epipolar lines, the wrong match can still be found if the scene contains one-dimensional or repetitive texture, and outright failure can

	Images	Unique images	Points	Average error / drift rate (pixels)	Maximum error / drift rate (pixels)
rover	101	51	55	13.8 / 0.14	51.6 / 0.52
highbay	27	14	13	3.82 / 0.15	14.8 / 0.57
helicopter	29	15	18	1.53 / 0.055	2.85 / 0.10
crane	39	20	77	1.65 / 0.043	36.94 / 0.97
arm	41	21	43	1.58 / 0.040	16.0 / 0.40

Table 6.5: Average and maximum errors for the synthetic symmetric sequences.

occur if one-dimensional or repetitive texture fills the field of view. Adapting multibaseline stereo techniques to sparse feature tracking would be an efficient way to increase the tracker accuracy and survivability in these situations.

2. The work on six degree of freedom motion estimation in Chapters 5 and 6 indicates that in some situations image-only estimation cannot reliably recover the camera motion. The same reasoning applies to the two- and three-frame motion estimation used within the tracker. So, understanding the tracker's behavior in these situations, and the possible benefits of integrating inertial observations into the tracker's motion estimation, are promising directions for increasing the robustness of the tracker. Specifically, gyro rather than accelerometer measurements are likely to be useful in the two- and three-frame motion estimation problems encountered in the tracker, because several images are likely to be needed to estimate the additional unknowns required to interpret accelerometer observations.

3. Previous chapters have argued that improved camera models and calibration could improve motion estimation, especially in the case of wide field of view cameras. Such work could also improve the performance of the tracker, and the current tracker implementation and test sequences would be useful as a starting point and stress test for such camera models.

The tracker's speed can also be improved. The current implementation is proof of concept, and requires roughly 20 seconds per image assuming a 1.8 GHz processor. The main steps needed to bring the system's performance up to real time are listed below, and as described below, systems that perform each of these *individual* capabilities in real-time have already been demonstrated.

1. Investigate David Nister's approach to performing real-time RANSAC for 2- and 3-frame motion estimation, and use these methods to make the tracker's epipolar estimation and geometric consistency RANSAC procedures real-time. These RANSAC procedures are the dominant computational expense, so this is the highest priority.

2. Optimize the radial distortion correction, the discrete epipolar line search, and the subpixel epipolar line search to achieve real-time performance. Real-time stereo systems have been demonstrated that perform radial distortion correction and rectification, and dense tracking of the features along epipolar lines in real-time. Tracking along epipolar lines in image streams differs from real-time stereo in that rectification of adjacent images in the video stream, to align the epipolar lines with raster lines, is not practical for general camera motion. However, tracking in the context of motion from images requires sparse rather than dense tracking, so rectification is probably not needed to achieve real-time performance.

3. Implement real-time SIFT feature extraction and matching. Real-time SIFT feature extraction and matching has already been demonstrated in a commercial product.

Chapter 7

Long-term motion estimation

7.1 Overview

As discussed in the previous chapters, existing systems are not suitable for the long-term motion estimation problems that arise in autonomous vehicle navigation and modeling from video, because they are susceptible to both gross local errors and long-term drift. Chapters 2-6 have investigated omnidirectional images, image and inertial measurements, and constrained tracking. Although the addition of an accelerometer to image-based motion estimation can provide two components of the sensors' orientation without drift, the methods described in these previous chapters are primarily concerned with reducing gross local errors in the estimated motion.

This chapter considers the problem of long-term drift directly. Limiting drift in cases where the camera revisits a previously mapped location requires:

- Recognizing from the images that a previously mapped location has been revisited.
- Using observations of the revisited location to reduce drift in the estimated positions.

The second problem has been extensively studied in the SLAM community, whereas recognizing a previously mapped location from images is unique to image-based motion estimation.

This chapter describes a proof of concept system for long-term motion that addresses these two problems, by exploiting the tracker described in the previous chapter, the variable state dimension filter (VSDF), and SIFT keypoints. This system is described in detail in Section 7.2 below, and the system's performance on a long image sequence in which the camera repeatedly revisits locations is described in Section 7.3. Section 7.4 discusses the remaining issues.

7.2 Method

Like many systems for estimating motion from images, the system estimates the current camera position using two steps. When a new image arrives, features are first tracked in the new image using the tracker described in Chapter 6. The resulting feature locations are then passed to a variable state dimension filter (VSDF) that estimates the camera position at the time of the new image, and the three-dimensional positions of the tracked points.

However, the system may also be able to generate a motion estimate for the current camera position with lower error by recognizing a previously mapped location, and extending an archived *rollback state* corresponding to that location. The following subsections describe this process in detail.

The variable state dimension filter, used for six degree of freedom motion estimation in the system, is a hybrid batch-recursive algorithm, described in detail by McLauchlan[38] in the context of shape and motion estimation from image measurements. Chapters 2 and 4 above have already suggested extending the VSDF for estimating sensor motion from image and inertial measurements. However, the VSDF used here is an image-only filter, extended to identify mistracked features using RANSAC.

SIFT keypoints are an enabling technology for recognizing previously mapped locations. SIFT keypoints are reviewed briefly in Section 2.6, and described in detail by Lowe[36].

7.2.1 Active state

An *active state* S_i describes the system after processing the most recent image, with index i. The structure of the active state is shown in Figure 7.1. The active state contains a list of image indices, I. During the system's initial operation, I will be the set of sequential indices $0, \ldots, i$. As we'll see below, however, I is not generally this sequential set.

The active state also contains an instance of the tracker described in Chapter 6. This tracker instance gives a history of sparse, two-dimensional feature locations found by tracking through the image sequence I. Similarly, the active state contains an instance of the variable state dimension filter that includes six degree of freedom position and error covariance estimates for the camera at the time of each image in I, and estimates of the three-dimensional point positions of the features that were tracked through the images in I. Last, the active state also contains the SIFT keypoints extracted in the most recent image, as shown in the figure.

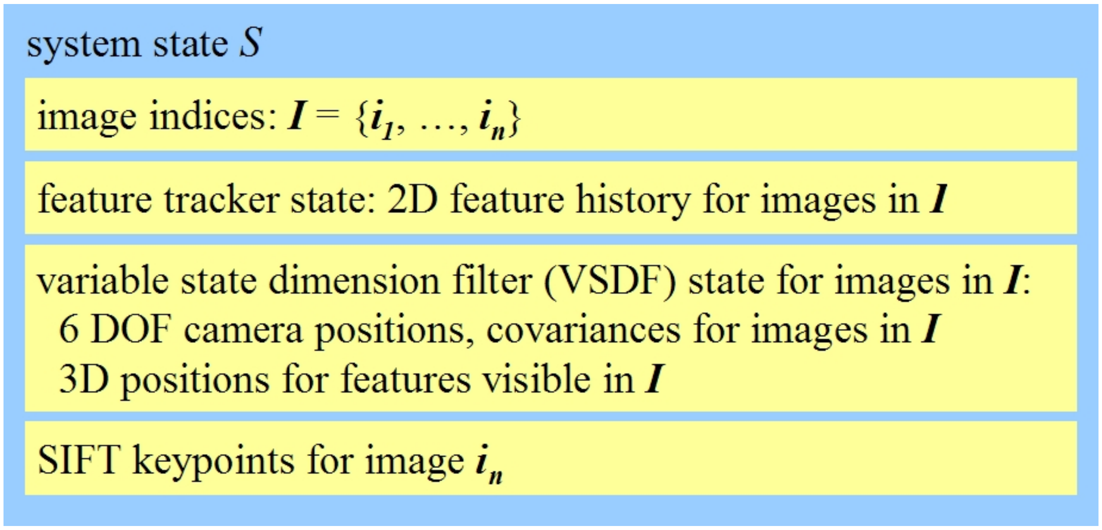


Figure 7.1: The structure of active state described in subsection 7.2.1 and the rollback states described in subsection 7.2.2.

7.2.2 Rollback states

The system also maintains a collection of *rollback states*, S_{r_0}, S_{r_1}, $\ldots$, that describe the system state at previous times. As described in the following subsections, maintaining these rollback states allows the system to revert to, and extend, a previous motion estimate if the camera revisits a previously mapped location. The structure of the rollback states is the same as the structure of the active state described in the previous section. In the experiments described below in Section 7.3, the archived rollback states are the states for every tenth image of the sequence.

7.2.3 Extending states

When a new image arrives, a new state S' can be generated by extending a previous state S in the obvious way:

- I' is constructed by appending the index of the new image to I.
- The tracker is copied from S into S' and extended by tracking into the new image.
- The VSDF is copied from S into S' and extended using the tracking data for the new image generated in item 3 above.

- Any features identified as outliers in the new image by the VSDF are eliminated from both the tracker and the VSDF, starting in the new image.
- SIFT keypoints are extracted in the image and included in S'.

For example, this method is used to generate a tentative new active state $S_{i-1,i}$ from the previous active state S_{i-1}, or as described below, to generate a tentative new active state $S_{j,i}$ from an archived rollback state S_j.

7.2.4 Operation

When a new image with index i arrives:

1. A candidate active state $S_{i-1,i}$ is created from the new image and S_{i-1}, as described in the previous subsection.
2. A cueing procedure, described below, is used to find a set of candidate rollback images, with indices $c_0, \ldots, c_k$.
3. Candidate active states $S_{c_0,i}, \ldots, S_{c_k,i}$ are created from the new image and the candidate rollback states $S_{c_0}, \ldots, S_{c_k}$.
4. The candidate state among $S_{i-1,i}, S_{c_0,i}, \ldots, S_{c_k,i}$ with the lowest estimated camera translation covariance for image i is adopted as the new active state. To determine the lowest of two covariances, the largest principal components of the covariances are compared.
5. If i is a multiple of n, the new state S_i is recorded for future use as a rollback state.

The cueing procedure in step 2 first identifies the rollback states whose most recent camera translation covariance estimate is lower than the camera translation covariance estimate for image i in the candidate new state $S_{i-1,i}$. As in step 4 above, the lowest of two covariances is determined by comparing the covariances' largest principal components. The rationale here is that extending rollback states whose most recent camera covariance is already larger than the current candidate's camera covariance is unlikely to result in a smaller covariance.

The cueing procedure then thresholds the rollback states identified in the previous step by computing the number of SIFT matches between the most recent image in the rollback state and image i. States with SIFT matches greater than some threshold are returned as rollback candidates.

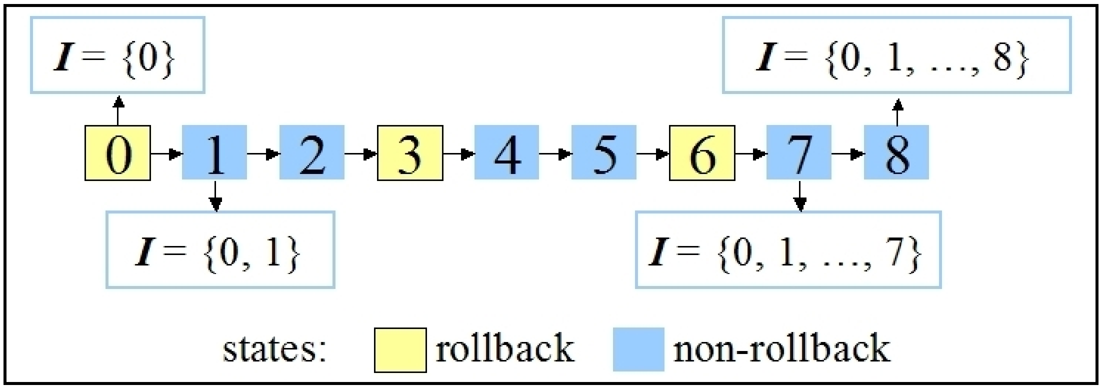


Figure 7.2: During initial operation, before any rollback state has been extended and adopted as an active state, the image index set for each state is $I = \{0, \ldots, i\}$.

7.2.5 Example

Figures 7.2-7.6 illustrate the system's operation. In Figure 7.2, images 0-7 have been processed without extending any rollbacks states, producing eight states with image index sets, $\{0\}$, $\{0, 1\}$, . . ., $\{0, 1, \ldots, 7\}$. When image 8 arrives, a tentative new active state $S_{7,8}$ is generated with images $\{0, 1, \ldots, 8\}$ by extending the previous active state S_7 into image 8. In this example, every third state (states S_0, S_3, and S_6) has been recorded as a rollback state, as shown in the diagram.

As shown in Figure 7.3, this results in both position and covariance estimates for image 8. In the proof of concept system, joint covariance estimates for the camera and point positions can be found by inverting the approximate Hessian matrix that the VSDF generates during the iterative minimization. A covariance estimate for a particular camera position or camera translation can then be found by marginalizing this joint covariance matrix.

Then suppose that the camera covariance for image 3 in state S_3 is lower than the camera covariance for image 8 in the tentative new active state $S_{7,8}$, as shown in Figure 7.3. A tentative new active state $S_{3,8}$ is then generated by extending the rollback state S_3 into image 8. If the resulting camera position for image 8 is found to have smaller covariance than the camera position estimate for image 8 in $S_{7,8}$, the tentative state $S_{7,8}$ is pruned and the extended state $S_{3,8}$ is adopted as the new active state, with image indices $\{0, 1, 2, 3, 8\}$. This is shown in Figures 7.4 and 7.5.

The situation after several more images have been processed is shown in Figure 7.6. In this example, a state generated by extending a rollback state is adopted as the new active state, when rollback states S_3, S_6, S_9, S_{12} are extended into images 8, 11, 14, and 17,

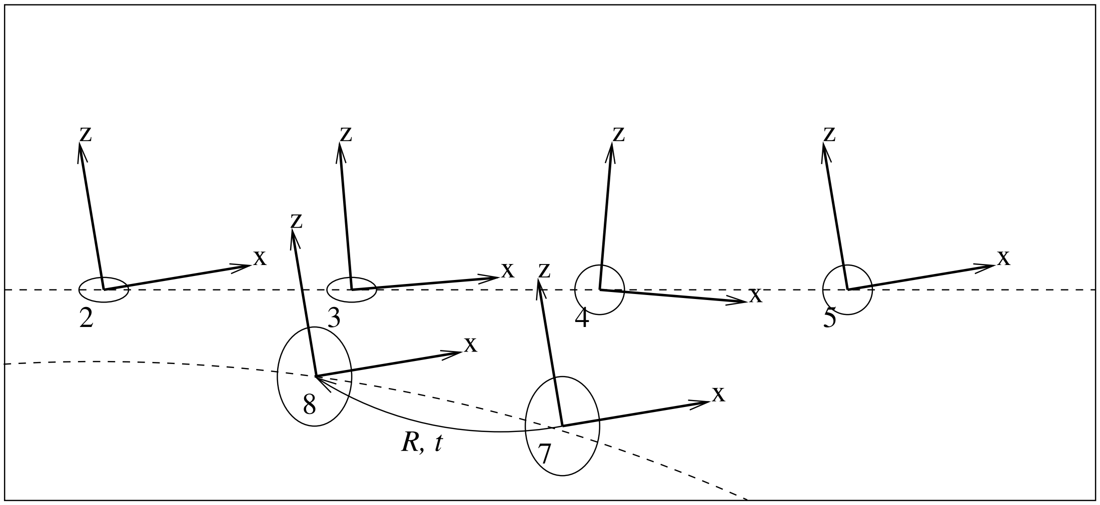


Figure 7.3: A new position and covariance are found during normal, non-rollback operation.

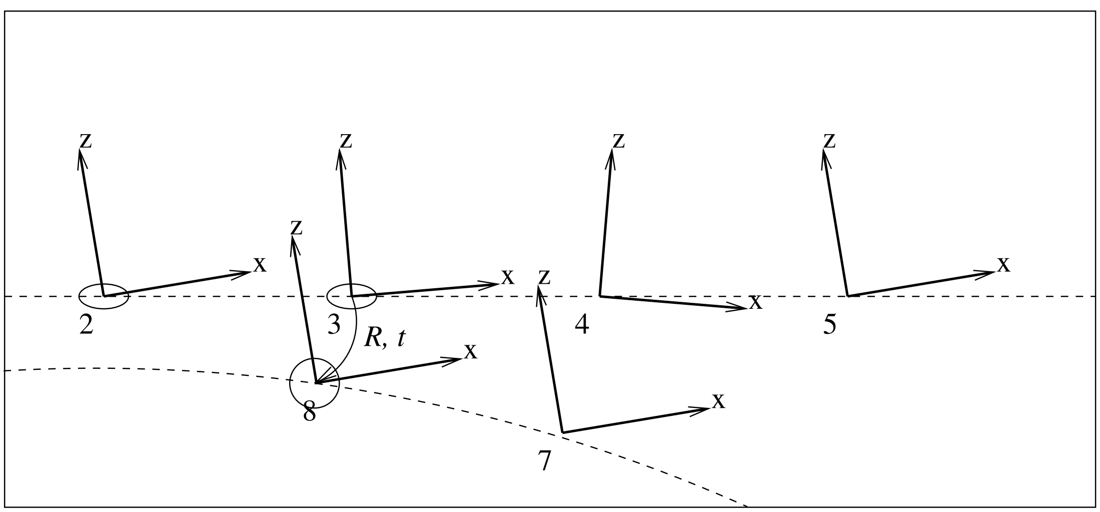


Figure 7.4: Extending a rollback state results in a new position estimate with a lower covariance than not rolling back. As the diagram shows, estimating a new camera position by extending a rollback state does not require that the camera revisit the exact location of a rollback camera position, and does not "snap" the new camera position to the old position.

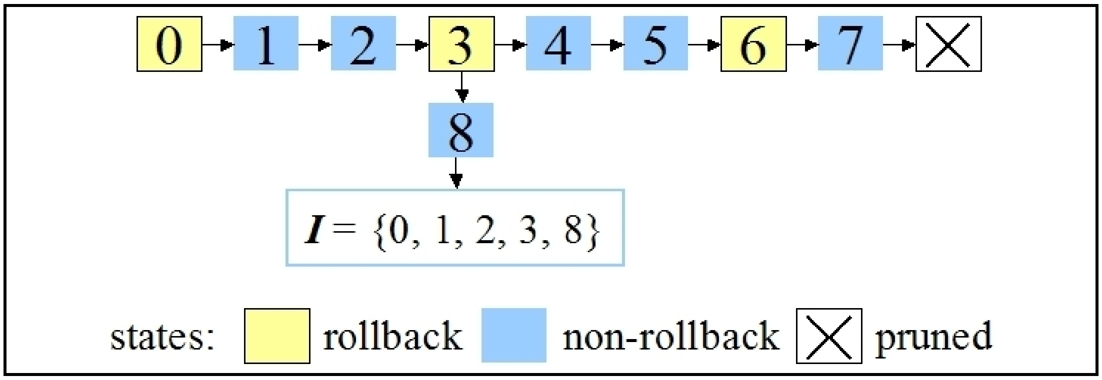


Figure 7.5: The state that results from extending a rollback state is adopted as the new active state, and the state that results from extending the previous active state is discarded.

respectively. The final active state that results, S_{20}, has been generated using images {0, . . ., 6, 11, 12, 17, . . . 20}.

7.3 Results

7.3.1 Overview

This section briefly describes the operation of the system on an extended, 945-image version of the "highbay" sequence described in Chapter 6. Subsection 7.3.2 describes this sequence in more detail. Subsection 7.3.3 describes the system parameter choices, and subsection 7.3.4 describes the resulting estimates.

7.3.2 Input sequence

The full "highbay" sequence is a 640×480, 945-image sequence taken from an IEEE 1394 camera with a wide field of view lens. The camera was mounted on a manually controlled cart, and makes three forward and three backward passes through a cluttered scene. The camera motion is planar is x, y and contains rotation about the camera's y (image up) axis.

A few representative images from the first forward pass, which occurs from image 0 to image 213, are shown in Figure 7.7. As shown in the images, the scene is highly non-planar, and the resulting images include severe occlusions, large image motions, and changes in overall intensity. Most areas are well-textured, but many parts of the image sequence contain poor texture (e.g., the saturated areas in images 7.7(b), 7.7(e), 7.7(f), and

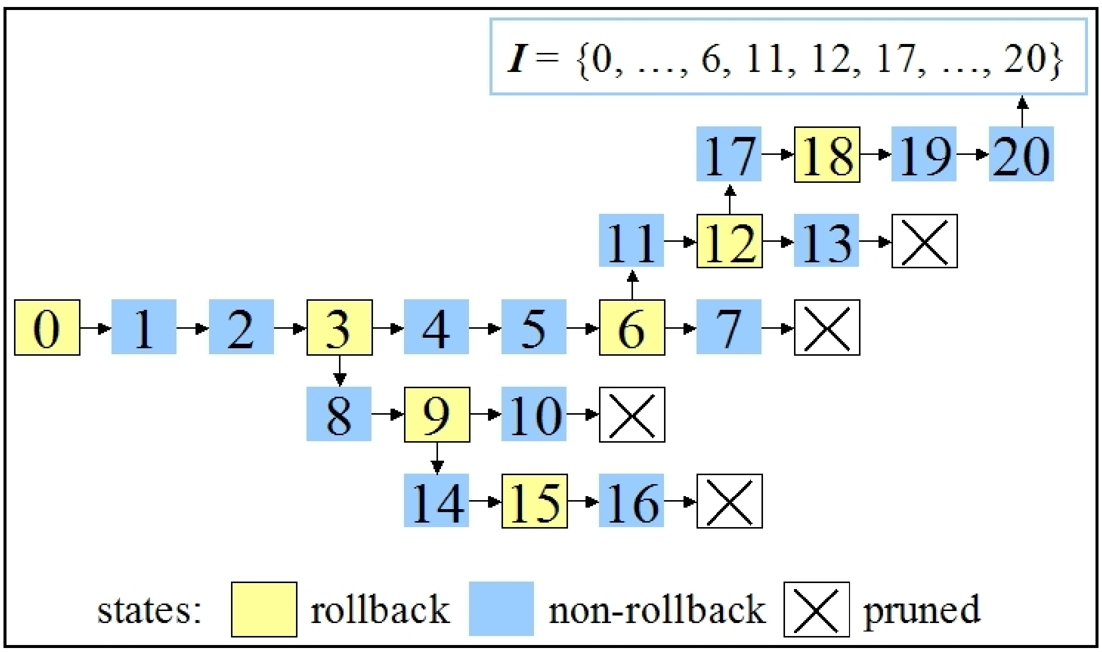


Figure 7.6: If several states that result from extending a rollback data have been adopted as the active state while processing the sequence, the current state and camera estimate may have been generated from a small number of images.

7.7(h)), repetitive texture (e.g., the oblique wall and calibration target in 7.7(f)), and one-dimensional texture (e.g., the overhead door in 7.7(h)).

Two estimates of the x, y camera motion during the first forward pass and the x, y sparse scene structure are shown Figure 7.8. These estimates were generated using the proof of concept system with two different sets of system parameters, and are described in more detail in the following subsections. The estimated locations of the camera at the time of the images (a)-(h) shown in Figure 7.7 are marked on the diagrams.

The motion during the subsequent passes traverses the same area, but may or may not include the visits to the side locations marked as "a", "c", and "e".

As shown in Figure 7.8, most of the imaged scene is on one side of the camera path. This is because the camera is mounted at a $45°$ angle, pointing forward and to the left of the cart.

7.3.3 System parameters

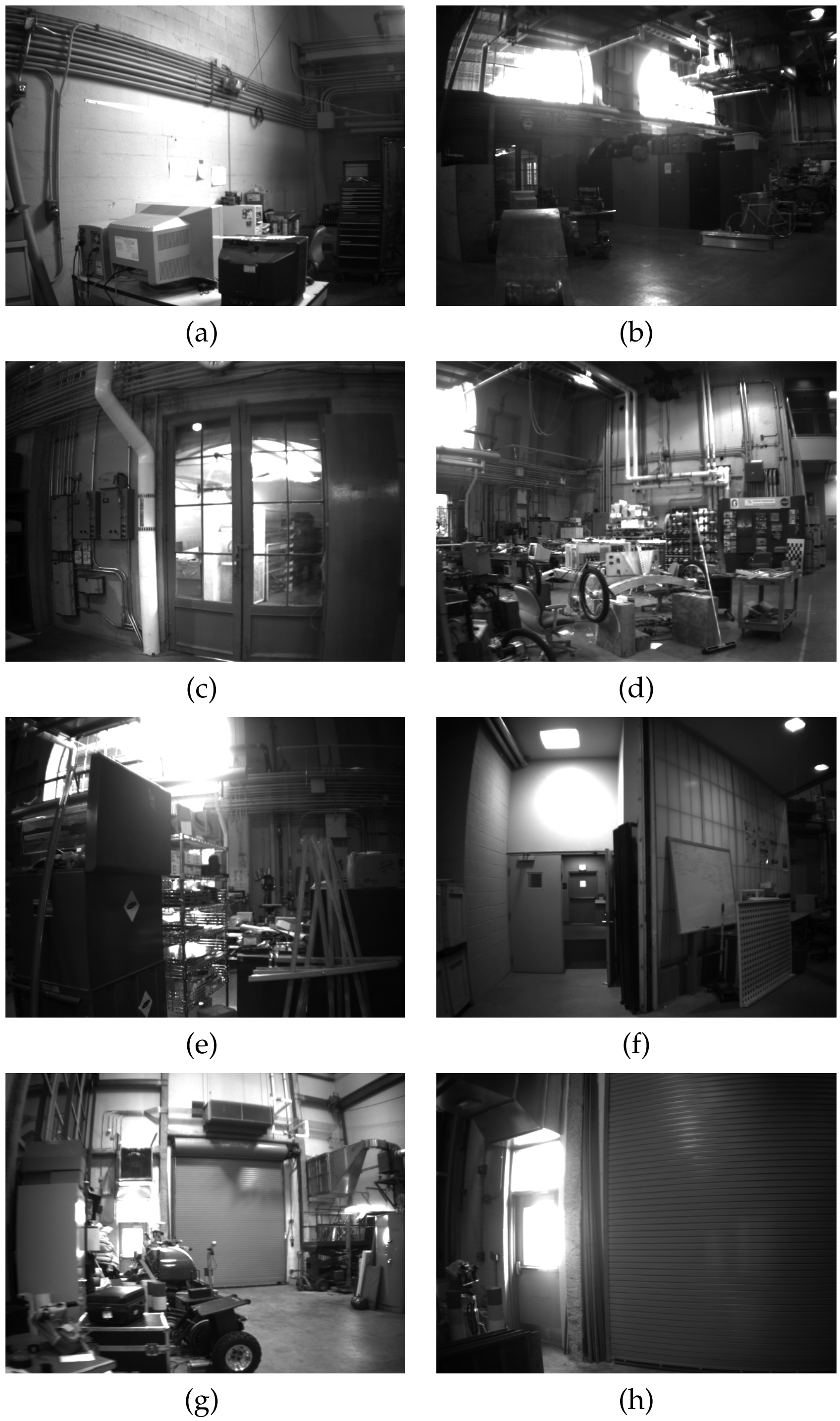
Two estimates of the motion and scene structure were generated using two different sets of system parameters. Most of the system parameters were the same in both experiments, including:

- The number of VSDF initialization images and the number of images in the VSDF sliding window are both 10.
- The number of SIFT matches needed to reacquire (i.e., to decide that the current image matches a previous image) is 300. This is the SIFT matches threshold described in the cueing step at the end of subsection 7.2.4.
- The number of RANSAC trials for both the tracker's geometric mistracking step and the VSDF's mistracking detection step were 500.

However, two of the parameters that most affect the overall system performance differ between the two datasets:

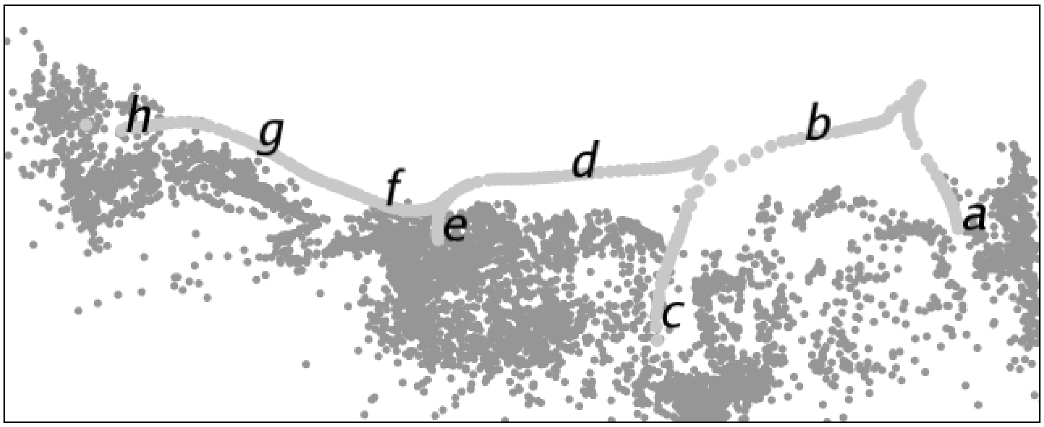
- The tracker's geometric mistracking RANSAC threshold was 2.0 pixels in the first experiment (Figure 7.8(a)) and 1.0 pixel in the second experiment (Figure 7.8(b)).
- Similarly, the VSDF's mistracking RANSAC threshold was 2.0 pixels in the first experiment (Figure 7.8(a)) and 1.0 pixel in the second experiment (Figure 7.8(b)).

Normally, the proof of concept system may generate states by rolling back to, and extending, a state that was itself generated by rolling back to and extending a previous state.

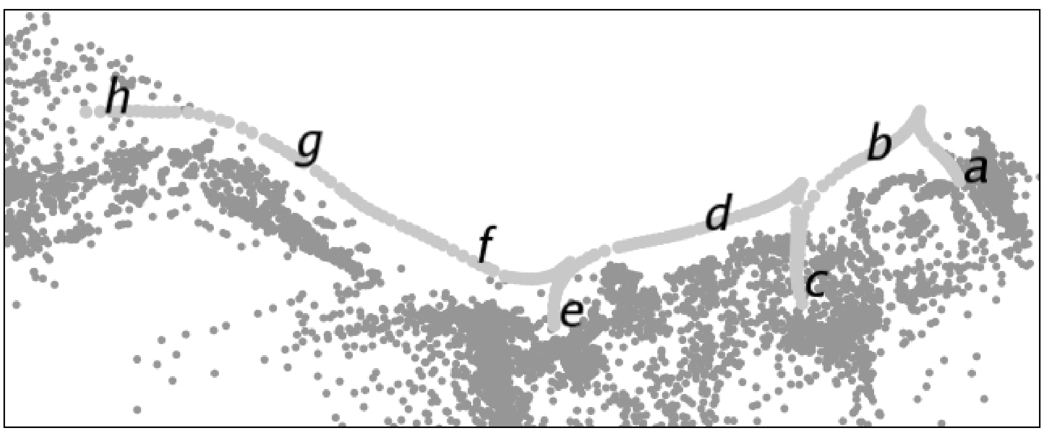


(a) (b) (c) (d) (e) (f) (g) (h)

Figure 7.7: Eight representative images from the first pass in the "highbay" sequence are shown in (a) through (h). These are images 0, 43, 65, 110, 138, 167, 191, and 212, respectively.



(a)



(b)

Figure 7.8: Two estimates of the "highbay" camera motion and sparse scene structure during the first forward pass, generated by the proof of concept system using two different sets of system parameters, are shown in (a) and (b). The estimated camera locations at the time of the images (a)-(h) in Figure 7.7 are marked on each diagram. The estimates and the system parameters used to generate them are described in detail in subsections 7.3.3 and 7.3.4.

That is, the relationships between parent and child states form an arbitrary tree. In this example, however, the camera is known to map the entire area during the initial forward pass spanning the first 214 images of the sequence. So, for clarity of presentation here, rolling back is suppressed during the first 214 images, and in the remaining 721 images, rolling back is limited to states recorded during the initial, 214 image pass. As a result, the overall accuracy of the estimates during the 945 sequence is largely determined by accuracy of the recovered motion for the initial 214 images.

7.3.4 Estimates

In both experiments, the system generates motion estimates with limited drift by correctly recognizing previously visited locations, and rolling back to the corresponding, previously recorded states. However, the estimates differ in several other respects.

In the first result, sufficient features are tracked throughout the entire initial pass to generate an accurate map of the area. As a result of the generous outlier thresholds, however, states that result from rolling back and extending previous states sometimes differ significantly from the motion indicated by the initial pass estimates.

In the second result, sufficient features are tracked throughout most of the initial pass. However, there is a noticable change in overall scale before and after location "f". However, rollbacks occur with more frequency and states that result from rolling back and extending previous states are highly consistent with the initial pass estimate.

The rollbacks for each experiment are summarized in Table 7.1. In the table, "$x \leftarrow y$" indicates that starting at image x, the rollback state corresponding to image y was reinstated and extended. Some example rollback image pairs from the third backward pass are shown in Figure 7.9.

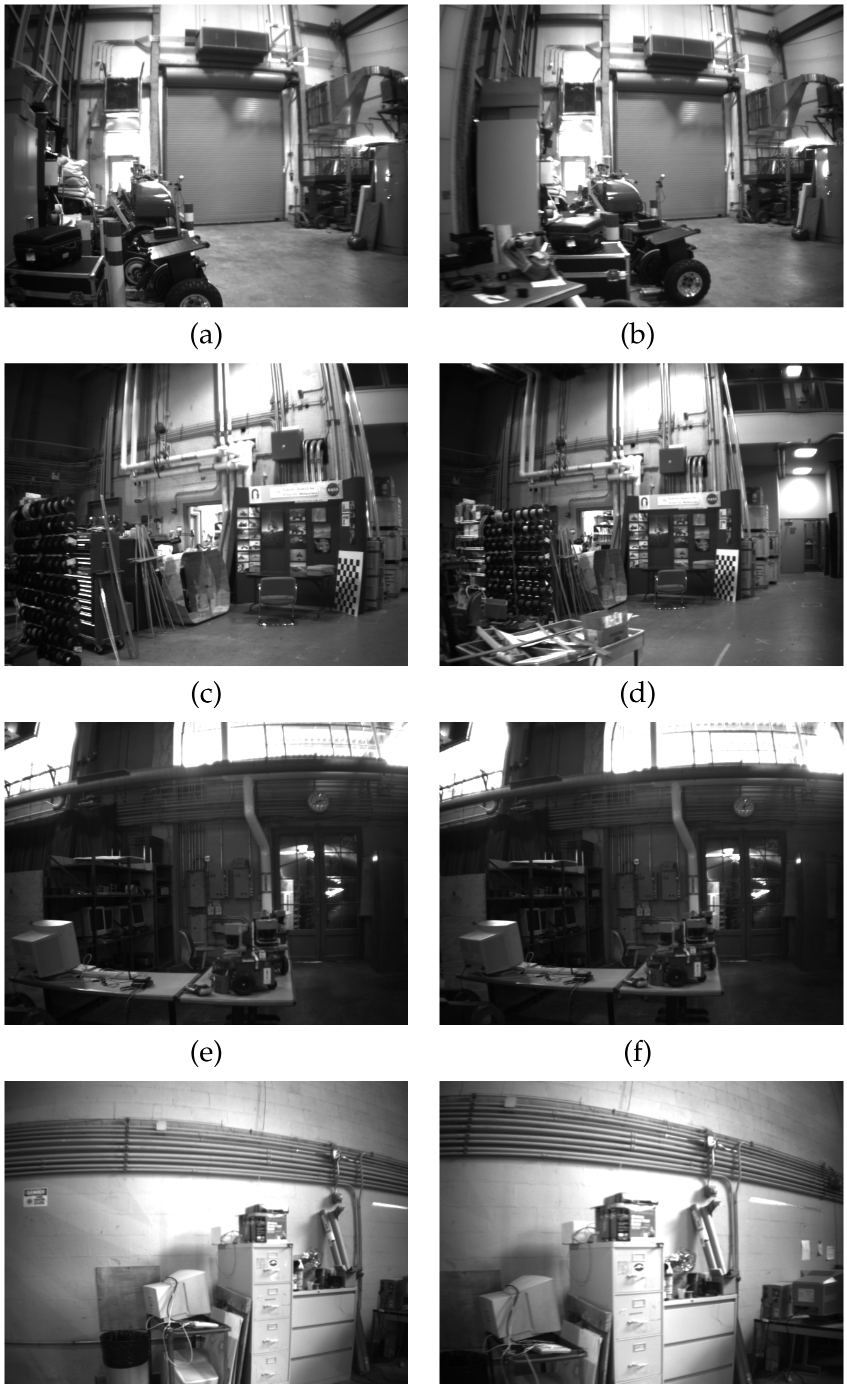
During the two experiments, the system loses the position estimate four times due to repetitive or one-dimensional texture. The corresponding four images are shown in Figure 7.10. After the position estimate is lost, the uncertainty in the position estimate is set to infinity, which allows the system to recover by rolling back the first time that the number of SIFT matches between the current image and a rollback image are above the 300 match threshold.

7.4 Discussion

The system described in this chapter performs long-term motion estimation by exploiting the tracker described in Chapter 6, the variable state dimension filter, and SIFT keypoints.

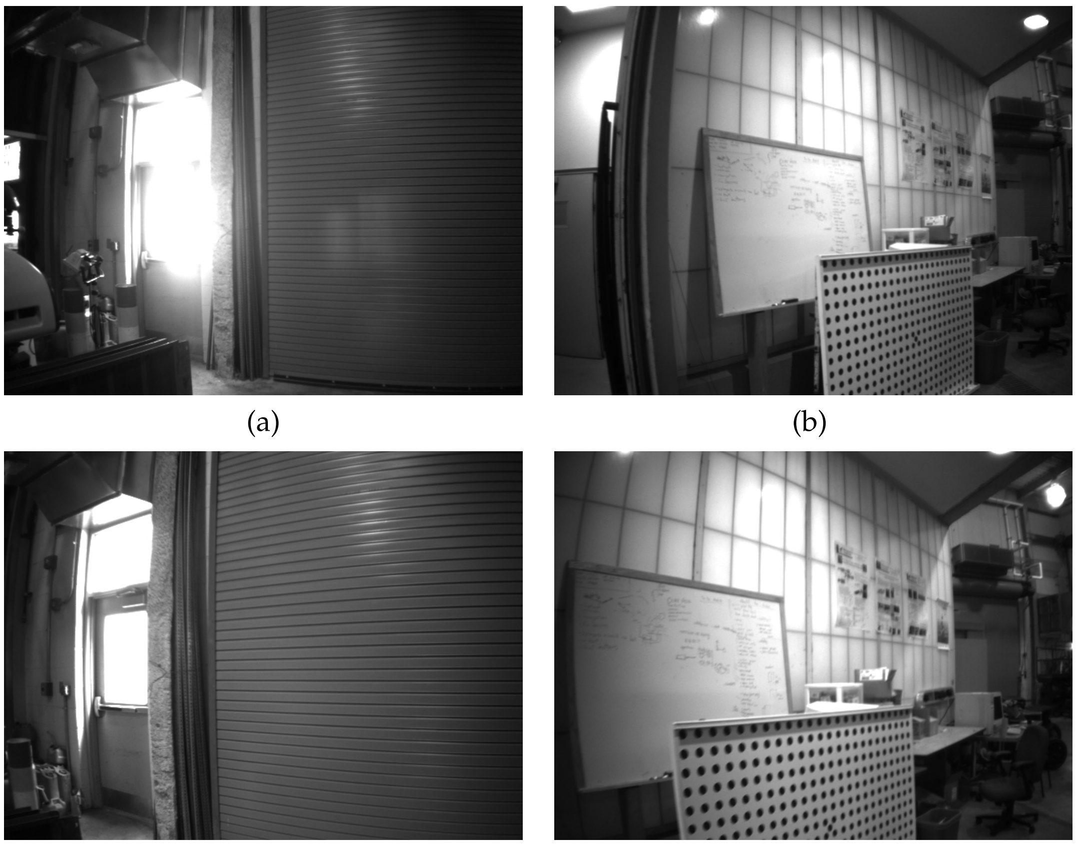
	2.0 pixel thresholds	1.0 pixel thresholds
First backward pass: (Images 214-380)	228 ← 200 308 ← 110 374 ← 040	228 ← 200 278 ← 160 301 ← 110 341 ← 060 343 ← 070 374 ← 040
Second forward pass: (Images 381-493)	416 ← 110 426 ← 120 433 ← 150 442 ← 160	454: position lost 460 ← 180 471 ← 190
Second backward pass: (Images 494-609)	496: position lost 507 ← 200 586 ← 100 601 ← 040	494: position lost 507 ← 200 514 ← 190 551 ← 160 572 ← 110 573 ← 120 601 ← 040
Third forward pass: (Images 610-762)	701 ← 150 712 ← 160	678 ← 100 726: position lost 730 ← 180 742 ← 190 753 ← 200
Third backward pass: (Images 763-944)	851 ← 100 907 ← 040 934 ← 010	779 ← 190 823 ← 150 829 ← 120 837 ← 110 872 ← 080 907 ← 040 934 ← 010

Table 7.1: The proof of concept system's reacquisition behavior for the first and second experiments is shown in the first and second, column, respectively.



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 7.9: Image pairs from rollbacks 1, 3, 5, and 7 on the third backward pass of the 1.0 pixel threshold experiment. The pairs are 779 ← 190 (images a and b), 829 ← 120 (images c and d), 872 ← 080 (images e and f), 934 ← 010 (images g and h).



(a) (b)

(c) (d)

Figure 7.10: During the 2.0 pixel threshold experiment, the system loses the position estimate one time, on image 496, shown in (a). During the 1.0 pixel threshold experiment, the system loses the position estimate three times, on images 454, 494, and 726, which are shown in (b), (c), and (d), respectively.

The system includes both capabilities that such a system needs: recognizing previously visited locations and exploiting observations of previously visited locations for reducing the errors in estimation.

However, the approaches to these two capabilities are just a starting point, and many issues in deploying such a system for everyday use still remain. For instance, in the system as it is described here, the number of recorded rollback states grows linearly with the number of images in the sequence, and each recorded rollback state is large. This limitation might be mitigated by reducing the size of the individual recorded rollback states, by reducing the number of tracked points, by adopting a one-dimensional (i.e., depth) rather than a three-dimensional parameterization of the estimated feature positions, or by investigating a more space efficient invariant feature method to replace SIFT features.

However, in the existing system, cueing time increases with the linearly increasing number of rollback states, and reducing the size of the individual rollback states does not address this problem. To fix this more fundamental problem, two approaches are possible. First, methods to minimize the number of recorded rollback states while still providing adequate coverage of the explored area should be investigated. Second, more effective use of the current camera position and covariances estimate should be made to reduce the number of recorded rollback states that are examined in full.

One difficulty specific to the first capability, recognizing previously visited locations, is that SIFT features cannot be matched across severe changes in scale and view angle. Both the system and the experimental data described in this chapter would be excellent starting points for exploring additional approaches for efficiently recognizing landmarks across more severe changes in scale and viewpoint, such as perceptual grouping.

The second capability, exploiting the recognition of a previously visited location, is addressed in a straightforward way by this system, but the system does not include some capabilities that have been investigated in the SLAM literature. For instance, efficiently updating the history of estimates, rather than just the most recent estimate, is required for closing the loop and producing an accurate overall map. The system described in this chapter would be an excellent starting point for investigating these ideas in the context of motion from images.

As described in the paragraphs above, some additional capabilities should be investigated for inclusion in the system, but at the same time, the initial experiment described in Section 7.3 does not explore all of the system's existing capabilities, and more experiments should be performed with the existing system. For example, the results in Section 7.3 were generating using a strict threshold on the number of SIFT matches between the current image and rollback images, but this strict threshold limits rollbacks to those states that are very close in viewpoint to the current image, as shown in Figure 7.9. The system is designed to correctly achieve and exploit rollbacks from more severe changes in viewpoint, and the system's performance for different choices of this threshold parameter should be explored in the future.

As a final observation, the system described in this chapter achieves long-term motion estimation by adding relatively simple components to a robust feature tracker, the variable state dimension filter, and SIFT, and the quality of the resulting estimates is highly dependent on the quality of these components. Although the problems of tracking, six degree of freedom motion estimation, and invariant image features have long been the subject of research in the vision community, the performance of these subsystems is still the limiting factor in the performance of the system described here.

Chapter 8

Conclusion

8.1 Overview

This work has focused on developing the robustness and accuracy needed to bring image-based motion estimation to long-term motion applications, such as autonomous vehicle navigation and modeling from video. In particular, the work has focused on four areas:

- Modeling and calibration of noncentral omnidirectional cameras, and motion estimation from omnidirectional images using the resulting model.
- Exploiting the complementary nature of image and inertial measurements for robust motion estimation.
- Making the robustness and accuracy of sparse image feature tracking meet the stringent requirements of motion estimation from real sequences.
- Exploiting feature reacquisition to limit drift in long-term motion estimation.

Sections 8.2, 8.4, and 8.5 below summarize the technical conclusions that can be drawn from this work, the contributions of this work, and possible future directions, respectively.

8.2 Conclusions

Some of the key conclusions that can be drawn from this work are:

1. Motion estimation from omnidirectional images benefits from a precise camera model, including the relative misalignment between the camera and mirror, even if the misalignment is slight. On the other hand, given a calibration, it is possible to perform

accurate motion estimation even if the misalignment between the camera and mirror is severe.

2. One calibration image is sufficient to determine the misalignment between an omnidirectional camera's mirror and camera.

3. Motion from image measurements alone can be highly sensitive to both small errors in the observed projections and to the camera intrinsics calibration.

4. Motion estimation from image and inertial measurements can produce accurate estimates even in cases where the estimates from image or inertial measurements alone are poor. Measurements from inexpensive inertial sensors can serve to eliminate the sensitivity that can occur in image-only estimation, estimate the global orientation, and establish the global scale. Image measurements are beneficial in limiting the drift that occurs in integrating inertial sensors, and make it possible estimate the unknowns that are needed to interpret inertial sensor outputs without any a priori assumptions.

5. Using both image measurements and measurements from inexpensive inertial sensors primarily benefits local motion estimates, rather than long-term motion estimates. In the long term, accelerometers' dependence on gravity can estimate two components of the sensors' global orientation without drift, but inexpensive inertial sensors are unlikely to eliminate gradual drift in azimuth or translation.

6. Omnidirectional images can provide some of the benefits of using both conventional image and inertial measurements, such as decreased sensitivity to very sparse observations. Of course, some potential benefits that inertial sensors provide, such as recovery of the global scale and the global orientation, cannot be recovered using just omnidirectional images.

7. Although they cannot be used for real-time or "infinite" operation, batch algorithms are an important tool for motion estimation. Estimates from these algorithms are sometimes useful in their own right. In other applications, the batch estimate is important in understanding the best quality we can expect given a particular sensor configuration, vehicle motion, environment, and set of observations, and in measuring the inherent sensitivity of the estimate with respect to random observation errors.

8. In both image-only and image-and-inertial estimation, the iterated extended Kalman filter (IEKF) can be made to work for long-term camera motion estimation given suf-

ficiently accurate image measurements. However, (1) the IEKF can be highly sensitive to the camera motion propagation variances, and (2) assuming the state contains only the current camera position, the IEKF does not model the relative errors between adjacent camera positions.

9. Exploiting the rigid scene assumption and eliminating the heuristics often used with sparse feature tracking can produce highly reliable, dense feature tracking suitable for camera motion estimation.

10. The position of a camera moving between a number of locations over a long time period can be estimated with limited drift using feature reacquisition and relatively simple extensions to feature tracking and recursive motion estimation.

8.3 Recommendations

As described in Section 8.1, this work has examined four approaches to improving motion from image measurements. This section includes some comments on the relative merits of these approaches, and some explicit recommendations for exploiting them in combination.

Chapter 5 explores the relative advantages of motion from omnidirectional images and of motion from conventional images and inertial measurements, and in the experiments in that chapter, the advantages over conventional images alone only are roughly the same in each case. However, the advantages of omnidirectional cameras are more limited in practice:

- The potential strength of omnidirectional images for motion estimation largely results from scene points being visible in more of the image sequence as the camera changes its view angle. In the case of a camera that sees $360°$ in azimuth and $180°$ in elevation, and moves in a convex environment, every point is visible all the time. However, most omnidirectional cameras do not see $180°$ in elevation, and most environments are not convex and contain occlusions, so this advantage is limited in most cases. Furthermore, points may not always be trackable in the omnidirectional image even if they are visible, further limiting the number of images in which they are useful for motion estimation.

- Omnidirectional images are susceptible to overexposure. The sun causes saturation outdoors, and overhead lights cause saturation indoors.

- As mentioned above, most omnidirectional cameras do provide a $180°$ view in elevation. $90 - 140°$ might be more typical values. In this case, a wide field of view conventional camera that provides $70-80^\circ$ in elevation (i.e., $140-160^\circ$ total field of view) would provide many of the advantages of an omnidirectional camera with the convenience of a conventional projection model.

- Feature tracking is more difficult in omnidirectional images than in conventional images.

- Motion from omnidirectional images alone cannot recover the global scale or establish components of the rotation without drift, whereas image and inertial measurements together can.

In short, motion from wide field of view, conventional image measurements and inertial measurements is preferable to motion from omnidirectional images in practice.

Chapter 5 also illustrates that image and inertial measurements can reduce sensitivity to tracking and camera calibration errors, but initial experiments in which six degree of freedom motion is estimated using a wide field of view camera and the tracker described in Chapter 6 indicate that these also significantly increase accuracy. However, neither of these is ideal alone. Accurate tracking is still needed to perform motion from image and inertial measurements; and motion from dense, accurate tracking alone cannot provide the global scale or estimate components of the rotation without drift, even in the best case.

Feature reacquisition, as described in Chapter 7, provides an advantage that none of the other approaches provide, i.e., limiting drift in the case where the camera revisits previously mapped locations. However, even when using reacquisition the accuracy of the overall map is limited by the accuracy of raw motion estimation when no revisition occurs and a loop cannot be closed. So, all efforts should be made to maximize the accuracy of motion estimation in the absence of reacquisition. In the context of the current system, this would entail incorporating inertial measurements into the existing, image-only system for long-term motion estimation described in Chapter 7.

8.4 Contributions

1. A general catadioptric projection model was developed, which can accommodate noncentral cameras and misalignment between the omnidirectional rig's camera and mirror. Existing batch and recursive algorithms for simultaneously estimating camera motion and sparse scene structure were extended to omnidirectional cameras using this projection model.

2. An algorithm was developed to determine the mirror-to-camera calibration required for item (1) from a single image of known three-dimensional points. It was shown that this calibration accurately recovers the mirror-to-camera transformation, allows accurate shape-from-motion and epipolar matching even when the camera and mirror are severely misaligned, and increases shape-from-motion and epipolar matching accuracy even when the camera and mirror are closely aligned.

3. A batch algorithm for estimating sensor motion and sparse scene structure and from image, gyro, and accelerometer measurements was developed, which requires no a priori assumptions or limits on the sensor motion or scene structure. The optimal estimates produced by this algorithm are useful in their own right, and as a gold standard for the comparison of online algorithms.

4. A recursive, IEKF-based algorithm for estimating sensor motion and sparse scene structure from image, gyro, and accelerometer measurements was developed. This algorithm is a multirate method, meaning that separate measurement update steps are used to incorporate the image and inertial measurements at the different rates at which they become available. The requirements for correctly initializing newly visible points in the IEKF framework were analyzed, and a method for initializing new points, adapted from the simultaneous localization and mapping (SLAM) community, was introduced.

5. Using a large suite of experimental results, this work has attempted to establish fundamental facts about motion estimation from image and inertial measurements, and to evaluate the performance of the specific algorithms described here.

6. An algorithm for tracking sparse image features for motion estimation was developed that greatly improves on the accuracy of Lucas-Kanade, the previous go-to tracker in the shape-from-motion community. This algorithm exploits the rigid scene assumption and SIFT keypoints, and eliminates the poor heuristics normally used for tracking sparse features.

7. A system was developed for simultaneously estimating the long-term motion of a camera that revisits points in a finite scene, and the sparse structure of that environment. In problems where the camera revisits previously mapped locations, this system is able to limit long-term drift.

8.5 Future directions

The system developed in this work would benefit from further work on a number of topics, which are described in this section. In each case, the existing system would provide an excellent starting point for investigation, and the experimental results from the existing system would set a high bar for evaluating new developments.

The IEKF-based recursive method for estimating motion from image and inertial measurements is limited by the inability to model the relative errors between adjacent camera estimates, the implicit assumption of motion smoothness, the approximation of the state estimate by a Gaussian, and the linearization of measurements around uncertain state estimates. As suggested in previous chapters, these problems can be eliminated or mitigated using the variable state dimension filter (VSDF), so a priority for future work should be the construction of a VSDF for estimating motion from image and inertial measurements based on the batch algorithm presented in this work.

Because the covariance between currently visible points and points that are no longer visible is not modeled in the current recursive algorithm, the recursive algorithm cannot improve previously estimated states based on observations of revisited points. This capability would be useful for closing the loop and improving the overall accuracy of motion estimates. Techniques for closing the loop from the simultaneous localization and mapping (SLAM) literature, which investigates this problem in the context of ground rovers with range sensors and odometry, would be a good starting point for this improvement.

The system described in Chapter 7 for long-term motion estimation uses a simple technique for recognizing previously visited locations that requires time linear in the number of images seen so far. More efficient techniques for fast recognition would be beneficial and the system developed in this work would be a good platform for evaluating such work.

The conventional camera model used in this work appears to be an important limiting factor in the accuracy of the system over the long term. Nonparametric and noncentral models for conventional cameras are a promising approach to increasing the accuracy of the camera model, and would also allow very-wide-angle conventional cameras (i.e., conventional cameras with fields of view near $180°$) to be modeled and incorporated into this system.

The accuracy of the inertial sensors used in this work appears to be limited by the assumed sensor models rather than the random noise in the sensor outputs. The batch algorithm for motion from image and inertial measurements developed in this work would be an excellent platform for developing and evaluating more sophisticated models of these sensors' behavior.

Like previous work in shape-from-motion, this work has assumed that the observation errors between projections for the same point in different images, and for different points in the same image, are uncorrelated. However, this is clearly not the case. For instance, the use of a common epipolar geometry for constraining the projections of different points in the same image introduces related errors in these projections. On the other hand, image-to-image tracking introduces a drift in projections for the same point in different images over time. Modeling the correlation between projection errors and efficiently incorporating the resulting model into motion estimation are both challenging problems that could be investigated in the context of this work.

Bibliography

[1] Simon Baker and Shree K. Nayar. A theory of single-viewpoint catadioptric image formation. *International Journal of Computer Vision*, 35(2):1–22, 1999.

[2] Yaakov Bar-Shalom and Xiao-Rong Li. *Multitarget-Multisensor Tracking: Principles and Techniques*. YBS Publishing, Storrs, Connecticut, 1995.

[3] J. M. Bradshaw, M. Sierhuis, Y. Gawdiak, R. Jeffers, N. Suri, and M. Greaves. Adjustable autonomy and teamwork for the personal satellite assistant. In *IJCAI-01 Workshop on Autonomy, Delegation, and Control: Interacting with Autonomous Agents*, Seattle, Washington, August 2001.

[4] Jeffrey M. Bradshaw, Maarten Sierhuis, Yuri Gawdiak, Hans Thomas, Mark Greaves, and William J. Clancy. Human-centered design for the personal satellite assistant. In *International Conference on Human-Computer Interaction in Aeronautics*, Toulouse, France, 2000.

[5] T. J. Broida, S. Chandrashekhar, and R. Chellappa. Recursive 3-D motion estimation from a monocular image sequence. *IEEE Transactions on Aerospace and Electronic Systems*, 26(4):639–656, July 1990.

[6] Ted J. Broida and Rama Chellappa. Performance bounds for estimating three-dimensional motion parameters from a sequence of noisy images. *Journal of the Optical Society of America A*, 6(6):879–889, June 1989.

[7] Michael J. Brooks, Wojciech Chojnacki, Darren Gawley, and Anton van den Hengel. What value covariance information in estimating vision parameters? In *Eighth IEEE International Conference on Computer Vision (ICCV 2001)*, Vancouver, Canada, 2001.

[8] J. S. Chahl and M. V. Srinivasan. Reflective surfaces for panoramic imaging. *Applied Optics*, 36(31):8275–8285, 1997.

[9] Lin Chai, William A. Hoff, and Tyrone Vincent. Three-dimensional motion and structure estimation using inertial sensors and computer vision for augmented reality. *Presence*, 11(5):474–491, 2002.

[10] John J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison-Wesley, Reading, Massachusetts, 1989.

[11] Matthew Deans and Martial Hebert. Experimental comparison of techniques for localization and mapping using a bearing-only sensor. In Daniela Rus and Sanjiv Singh, editors, *Experimental Robotics VII*, pages 395–404. Springer-Verlag, Berlin, 2001.

[12] Matthew C. Deans. *Bearings-only localization and mapping*. PhD thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania, 2002.

[13] S.F. El-Hakim. 3D modeling of complex environments. In *SPIE Proceedings, Electronic Imaging 2001, Volume 4309: Videometrics and Optical Methods for 3D Shape Measurement VII*, San Jose, California, January 2001.

[14] Eric Foxlin and Leonid Naimark. VIS-Tracker: A wearable vision-inertial self-tracker. In *IEEE Virtual Reality Conference (VR 2003)*, Los Angeles, March 2003.

[15] Eric M. Foxlin. Generalized architecture for simultaneous localization, auto-calibration, and map-building. In *IEEE/RSJ Conference on Intelligent Robots and Systems (IROS 2002)*, Lausanne, Switzerland, October 2002.

[16] Arthur Gelb, editor. *Applied Optimal Estimation*. MIT Press, Cambridge, Massachusetts, 1974.

[17] Christopher Geyer and Kostas Daniilidis. Catadioptric camera calibration. In *Seventh IEEE International Conference on Computer Vision (ICCV 1999)*, pages 398–404, September 1999.

[18] Christopher Geyer and Kostas Daniilidis. Catadioptric projective geometry. *International Journal of Computer Vision*, December 2001.

[19] Christopher Geyer and Kostas Daniilidis. Structure and motion from uncalibrated catadioptric views. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2001)*, December 2001.

[20] Christopher Geyer and Kostas Daniilidis. Paracatadioptric camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(5), May 2002.

[21] Christopher Geyer and Kostas Daniilidis. Conformal rectification of an omnidirectional stereo pair. In *Proceedings of the Omnidirectional Workshop and Sensor Networks (OMNIVIS 2003)*, Madison, Wisconsin, June 2003.

[22] Christopher Geyer and Kostas Daniilidis. Mirrors in motion: Epipolar geometry and motion estimation. In *Ninth IEEE International Conference on Computer Vision (ICCV 2003)*, Nice, France, 2003.

[23] Joshua Gluckman and Shree K. Nayar. Ego-motion and omnidirectional cameras. In *Sixth IEEE International Conference on Computer Vision*, pages 999–1005, Bombay, January 1998.

[24] Joel M. Grasmeyer and Matthew T. Keennon. Development of the Black Widow Micro Air Vehicle. In *39th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, January 2001.

[25] Chris Harris. Geometry from visual motion. In Andrew Blake and Alan Yuille, editors, *Active Vision*, pages 264–284. MIT Press, Cambridge, Massachusetts, 1992.

[26] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, United Kingdom, 2000.

[27] Janna Heikkilä and Olli Silvén. A four-step camera calibration procedure with implicit image correction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1997)*, pages 1106–1112, San Juan, Puerto Rico, 1997.

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

[29] Andreas Huster, Eric W. Frew, and Stephen M. Rock. Relative position estimation for AUVs by fusing bearing and inertial rate sensor measurements. In *Oceans 2002 Conference*, pages 1857–1864, Biloxi, Mississippi, October 2002.

[30] Andreas Huster and Stephen M. Rock. Relative position estimation for intervention-capable AUVs by fusing vision and inertial measurements. In *Twelfth International Symposium on Unmanned Untethered Submersible Technology*, Durham, New Hampshire, August 2001.

[31] Andreas Huster and Stephen M. Rock. Relative position estimation for manipulation tasks by fusing vision and inertial measurements. In *Oceans 2001 Conference*, volume 2, pages 1025–1031, Honolulu, November 2001.

[32] Andreas Huster and Stephen M. Rock. Relative position sensing by fusing monocular vision and inertial rate sensors. In *Eleventh International Conference on Advanced Robotics (ICAR 2003)*, volume 3, pages 1562–1567, Coimbra, Portugal, July 2003.

[33] Sang-Hack Jung and Camillo J. Taylor. Camera trajectory estimation using inertial sensor measurements and structure from motion results. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2001)*, volume 2, pages 732–737, Kauai, Hawaii, December 2001.

[34] Y. Kanazawa and K. Kanatani. Do we really have to consider covariance matrices for image features? In *Eighth IEEE International Conference on Computer Vision (ICCV 2001)*, Vancouver, Canada, July 2001.

[35] Sing Bing Kang. Catadioptric self-calibration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2000)*, volume 1, pages 201–207, Hilton Head Island, South Carolina, June 2000.

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

[37] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Seventh International Joint Conference on Artificial Intelligence*, volume 2, pages 674–679, Vancouver, Canada, August 1981.

[38] Philip F. McLauchlan. The variable state dimension filter applied to surface-based structure from motion. Technical Report VSSP-TR-4/99, University of Surrey, Guildford, UK, 1999.

[39] Michael Montemerlo and Sebastian Thrun. Simultaneous localization and mapping with unknown data association using FastSLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, Taipei, Taiwan, 2003.

[40] Toshiharu Mukai and Noboru Ohnishi. The recovery of object shape and camera motion using a sensing system with a video camera and a gyro sensor. In *Seventh IEEE International Conference on Computer Vision (ICCV 1999)*, pages 411–417, Corfu, Greece, September 1999.

[41] Shree K. Nayar. Catadioptric omnidirectional camera. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1997)*, pages 482–488, San Juan, Puerto Rico, June 1997.

[42] David Nistér. An efficient solution to the five-point relative pose problem. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, volume 2, pages 195–202, Madison, Wisconsin, June 2003.

[43] David Nistér. Preemptive ransac for live struture and motion estimation. In *IEEE International Conference on Computer Vision (ICCV 2003)*, pages 199–206, Nice, France, October 2003.

[44] David Nistér. An efficient solution to the five-point relative pose problem. *IEEE Transations on Pattern Analysis and Machine Intelligence*, 26(6):756–770, June 2004.

[45] David Nistér. A minimal solution to the generalised 3-point pose problem. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2004)*, volume 1, pages 560–567, Washington, D.C., June 2004.

[46] Mark Ollis, Herman Herman, and Sanjiv Singh. Analysis and design of panoramic stereo vision using equi-angular pixel cameras. Technical Report CMU-RI-TR-99-04, Carnegie Mellon University, Pittsburgh, Pennsylvania, January 1999.

[47] Conrad J. Poelman. *The Paraperspective and Projective Factorization Methods for Recovering Shape and Motion*. PhD thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania, July 1995.

[48] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, United Kingdom, 1992.

[49] Gang Qian and Rama Chellappa. Structure from motion using sequential Monte Carlo methods. In *Eighth IEEE International Conference on Computer Vision (ICCV 2001)*, pages 614–621, Vancouver, Canada, July 2001.

[50] Gang Qian, Rama Chellappa, and Qinfen Zhang. Robust structure from motion estimation using inertial data. *Journal of the Optical Society of America A*, 18(12):2982–2997, December 2001.

[51] C. Rasmussen and G.D. Hager. Probabilistic data association method for tracking complex visual objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):560–576, June 2001.

[52] Henrik Rehbinder and Bijoy K. Ghosh. Rigid body state estimation using dynamic vision and inertial sensors. In *Fortieth IEEE Conference on Decision and Control (CDC 2001)*, pages 2398–2403, Orlando, Florida, December 2001.

[53] Stephen Se, David Lowe, and Jim Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *International Journal of Robotics Research*, 21(8):735–758, 2002.

[54] Randall Smith, Matthew Self, and Peter Cheeseman. Estimating uncertain spatial relationships in robotics. In Ingemar J. Cox and Gordon T. Wilfong, editors, *Autonomous Robot Vehicles*, pages 167–193. Springer-Verlag, New York, 1990.

[55] Richard Szeliski and Sing Bing Kang. Recovering 3D shape and motion from image streams using nonlinear least squares. *Journal of Visual Communication and Image Representation*, 5(1):10–28, March 1994.

[56] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[57] Bill Triggs, Philip McLauchlan, Richard Hartley, and Andrew Fitzgibbon. Bundle adjustment – A modern synthesis. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, LNCS, pages 298–375. Springer Verlag, 2000.

[58] Roger Y. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1986)*, pages 364–374, Miami, Florida, 1986.

[59] Roger Y. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, RA-3(4):323–344, 1987.

[60] John (Juyang) Weng, Yuntao Cui, and Narendra Ahuja. Transitory image sequences, asymptotic properties, and estimation of motion and structure. *IEEE Transactions on Image Analysis and Machine Intelligence*, 19(5):451–464, May 1997.

[61] Paul R. Wolf. *Elements of Photogrammetry*. McGraw-Hill, New York, 1983.

[62] Suya You and Ulrich Neumann. Fusion of vision and gyro tracking for robust augmented reality registration. In *IEEE Virtual Reality Conference (VR 2001)*, pages 71–78, Yokohama, Japan, March 2001.