Bearings-Only Localization and Mapping

Matthew Charles Deans

September 15, 2005
CMU-RI-TR-05-41

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
M. Hebert, Chair
Sebastian Thrun
Anthony Stenz
M. W. M. Gamini Dissanayake, University of Sydney

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

This research was sponsored by the National Aeronautical and Space Administration (NASA) and the NASA GSRP Fellowship Program.

Keywords: localization, mapping, mobile robots, omnidirectional vision, non-linear state estimation, kalman filter, unscented filter, gaussian quadrature

Abstract

In many applications, mobile robots must be able to localize themselves with respect to environments which are not known *a priori* in order to navigate and accomplish tasks. This means that the robot must be able to build a map of an unknown environment while simultaneously localizing itself within that map. The so-called *Simultaneous Localization and Mapping* or SLAM problem is a formulation of this requirement, and has been the subject of a considerable amount of robotics research in the last decade.

This thesis looks at the problem of localization and mapping when the only information available to the robot is measurements of relative motion and bearings to features. The relative motion sensor measures displacement from one time to the next through some means such as inertial measurement or odometry, as opposed to externally referenced position measurements like compass or GPS. The bearing sensor measures the direction toward features from the robot through a sensor such as an omnidirectional camera, as opposed to bearing and range sensors such as laser rangefinders, sonar, or millimeter wave radar.

A full solution to the bearing-only SLAM problem must take into consideration detecting and identifying features and estimating the location of the features as well as the motion of the robot using the measurements. This thesis focuses on the estimation problem given that feature detection and data association are available. Estimation requires a solution that is fast, accurate, consistent, and robust.

In an applied sense, this dissertation puts forth a methodology for building maps and localizing a mobile robot using odometry and monocular vision. This sensor suite is chosen for its simplicity and generality, and in some sense represents a minimal configuration for localization and mapping.

In a broader sense, the dissertation describes a novel method for state estimation applicable to problems which exhibit particular nonlinearity and sparseness properties. The method relies on deterministic sampling in order to compute sufficient statistics at each time step in a recursive filter. The relationship of the new algorithm to bundle adjustment and Kalman filtering (including some of its variants) is discussed.

Contents

List of Figures

Notation

$\mathbf{m}$	trajectory (all vehicle poses)
$\mathbf{m_i}$	vehicle pose at time i
$\mathbf{s}$	map (position of all features)
$\mathbf{s_j}$	position of feature j
$\mathbf{x}$	all model parameters, i.e. the map and trajectory
$\mathbf{d_i}$	odometry measurement i
b_k	bearing measurement k
$\mathbf{z}$	vector of all measurements
$\mathbf{f}()$	odometry measurement function
$\mathbf{F}$	Jacobian of $\mathbf{f}$
$g()$	bearing measurement function
$\mathbf{G}$	Jacobian of $g()$
$\mathbf{h}()$	general measurement function
$\mathbf{H}$	Jacobian of $\mathbf{h}$
$\mathbf{w}$	odometry measurement noise
v	bearing measurement noise
$\mathbf{Q}$	covariance of odometry noise
R	covariance of bearing measurement noise
$E_p\{\}$	expectation computed over distribution p
J	cost function
∇J	gradient of cost function
$\nabla^2 J$	Hessian of cost function
$\mathbf{a}$	approximation to ∇J
$\mathbf{A}$	approximation to $\nabla^2 J$

Part I

Introduction

Chapter 1

Bearing only localization and mapping

An essential aspect of autonomy for a mobile robot is the capability to determine its location. This capability is known as *localization*. Localization is typically a prerequisite for accomplishing real tasks, whether it is exploration, navigation toward a known goal, transportation of material, construction or site preparation.

In many applications, the mobile robot has an *a priori* map. Given a map, the robot may localize by matching current sensor observations to features in the map. Given enough features and an unambiguous geometry, the pose of the robot can be determined or at least narrowed down to a set of possible locations.

Usable maps do not always exist, and it is not always possible to have accurate externally referenced pose estimates. Planetary rovers may enter areas that have not been mapped out previously, or have not been mapped at high enough resolution for localization on a small scale. Mining robots might operate in areas that change dramatically. Underwater robots often explore sites that have never been visited before. If an a priori map is not available, the robot may need to construct one. With a precise, externally referenced position estimate from GPS or similar means, the robot can take its sensor observations, reference the observations to its current pose, and insert features in the map in the appropriate places.

Without maps or externally referenced pose information, the robot must produce its own map and concurrently localize within that map. This problem has been referred to as *concurrent localization and mapping* (CLM) and *simultaneous localization and mapping* (SLAM). This thesis focuses on the SLAM problem in which the robot is equipped with a bearing-only sensor and odometry and computes a metric, feature-based map of its environment without a priori information about feature locations.

1.1 Motivation

The SLAM problem has been considered in many contexts. Frequently approaches rely on accurate range-bearing sensors such as laser range scanners, millimeter-wave radar, imaging sonar, or active stereo vision. Motion measurements often come from accurate inertial measurement systems, gyroscopes, etc. or even externally referenced sensors such as GPS or compass.

Below is a discussion of the motivation for many of the choices in sensors, models, and algorithms used in this thesis.

1.1.1 Monocular vision and odometry

Much of the existing work in localization and mapping uses range sensors such as laser rangefinders, millimeter wave radar, and sonar[10, 17, 20, 21, 30, 34, 39, 47, 49, 50, 59, 62, 79, 80, 85, 86, 87, 96]. The first two sensing modalities typically have much higher power and mass than a single camera. Furthermore, most sensors require mechanical scanning mechanisms which reduce the fault tolerance of the sensors and limit the data acquisition rates. Sonar sensors used in indoor applications typically have wide beam profiles, making it difficult to identify the origin of the return signal. Range is typically limited as well. Use of sonar on outdoor robots is far less common.

Monocular vision does not provide range information. Using a pair of stereo cameras, depth maps of a 3-D scene can be computed from a single pair of images with the appropriate processing. However, sets of cameras require more mass, power, volume, and hardware support (framegrabbers, etc.) compared to a single monocular camera. Furthermore, stereo processing can require a significant computational effort, and stereo vision is prone to its own problems. Monocular vision provides bearing information in the sense that each pixel in an image corresponds to a cone of light which passes through the focal center of the camera and the pixel location on the image plane. The depth information is lost due to the projection of the scene onto the image plane, but if images are combined from multiple poses, then the poses from which the images were taken as well as the locations of the features in the images can be determined. The low weight, low power, and low cost make monocular vision an attractive sensor choice for simple mobile robots. The sensor can also serve multiple purposes; onboard cameras are useful for visualization, remote science, teleoperation, etc.

Bearing only measurements are a difficult case for several reasons. First, a single measurement is insufficient for computing an estimate of the location of a landmark, since the measurement is merely a direction and the distance is unknown. This makes it difficult to initialize the location of a new landmark during mapping. Second, the nonlinear nature of the measurement makes it difficult to compute useful posterior statistics for recursive filtering methods such as Kalman filters, since the posterior statistics rely on linearization near the true state in order to be useful. Also, it will be argued later that without range measurements some other sensor must provide information about the overall scale of the map and the robot motion.

In addition to monocular vision, the work presented here makes use of odometry information. Odometry can provide the scale information which is missing in the bearing only measurement. There have been examples of robots localizing and mapping without odometry[50], but this requires range sensing which we do not assume to be available. Furthermore, odometry provides some prior information on the robot motion which can help to disambiguate the solution. Fitting robot motion to the odometry also tends to favor smooth robot motion since the kinematic and dynamic constraints of the vehicle will disallow erratic motion. This tends to regularize the problem.

This work does not consider the inclusion of external position references. There are many cases in which external heading or position references do not exist or are unreliable. Single antenna GPS receivers do not provide orientation information; gyroscopic compasses drift; magnetic compasses are only useful in environments where there is a useful magnetic field, e.g. not on Mars or near the Earth's poles. Even then magnetic compasses can exhibit limited slew rates, large noise, and bias in the presence of magnetic fields generated by the robot itself.

It should be noted that GPS, compass, gyroscopes, accelerometers and other sensors can be incorporated into this framework. The addition of these measurements will improve the state estimates, especially externally referenced ones which can correct drift.

1.1.2 Batch mode and Recursive estimation

Batch estimation techniques consider all of the available data at once in order to compute the best solution available given the data. Batch estimation techniques have the advantage that they treat all data with equal importance and find optimal estimates for all model parameters given all data. Batch estimation can also often serve as a "gold standard" method, since it can provide a way to compute the best solution available given the data, decoupling the errors that are due to the data itself, and the errors due to approximations or other characteristics of algorithms which are preferred for computational reasons.

However, full batch estimation solutions are slow due to the volume of data that a mobile robot can gather in even a fairly modest period of time. For this reason, we desire a recursive state estimation method so that measurements can be incorporated as they are made and later discarded.

Kalman filter based solutions to SLAM or SFM incorporate the data as it becomes available and replaces the data with a statistic. The statistic used is an estimated posterior mean and covariance. It is known that this estimate is biased for nonlinear problems and the statistic may not capture the true posterior well, causing the Kalman filter to drift or become inconsistent. The primary issues in computing statistics in a Kalman filter for bearing only SLAM are that estimating a landmark location with few measurements is ill posed, and that therefore computing linearizations after only a small number of measurements of a landmark are available means potentially linearizing at the wrong point. The posterior statistics do not capture the data.

The estimate can be improved if we use a nonlinear smoothing filter, since the measurements are linearized and summarized using a better (lower variance) state estimate. The incorporation of more information with which to compute state estimates improves the quality of the estimates. The estimate can also be improved if we use statistical linearization, since the posterior statistics are valid over the region of high probability rather than only at the point corresponding to the current state estimate. Given that we choose to represent the posterior state estimate as a Gaussian, this thesis will show that smoothing and statistical linearization can be used to compute better approximations for recursive estimation in SLAM.

1.2 Modeling the system

Recursive filters are typically described in terms of process and measurement models. The process model is a description of how the state evolves as a function of time. The measurement model is a description of what measurement would be expected for a given state. In general these models are nonlinear and only approximate the true nature of the system. The filter also requires a representation of the system state and measurements.

1.2.1 System state

In the work presented here, we model the robot environment as a 2-D planar world. The rover is modeled as a rigid body. The rover pose at time i is the 3-DOF (degree of freedom) parameter vector

$$\mathbf{m_i} = \begin{pmatrix} m_x \\ m_y \\ m_\theta \end{pmatrix} \tag{1.1}$$

which is the position of the rover on the 2-D plane and the orientation or yaw of the rover with respect to some conventional reference direction.

Landmarks are assumed to be point features. The vector describing landmark j is the 2-DOF parameter vector

$$\mathbf{s_j} = \begin{pmatrix} s_x \\ s_y \end{pmatrix} \tag{1.2}$$

which is the position of the landmark on the 2-D plane.

There are many rover and landmark attributes which are not included in this representation. The 2-D world model does not include elevation for the rover or landmarks. The rover attitude is not fully described, since the roll and pitch angles are not included. There is nothing in the sequel which precludes a 3-D model but generalizing the approach to 3-D would require a reparameterization. In particular the attitude of the rover is not trivial to encode. Euler angles or unit quaternions might be used but both approaches have their limitations. The

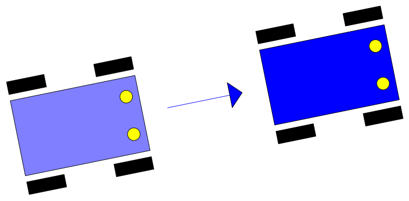


Figure 1.1: Vehicle odometry model

rover might also have articulation which is ignored here. Landmark descriptions could include many attributes such as size, shape, appearance, etc.

For notational convenience, we denote the vector of all unknown parameters

$$\mathbf{x} = \{\mathbf{m_1}^T, \mathbf{m_2}^T, \ldots, \mathbf{m_M}^T, \mathbf{s_1}^T, \mathbf{s_2}^T, \ldots, \mathbf{s_N}^T\}^T \tag{1.3}$$

The vector $\mathbf{x}$ includes the entire description of the state of the system subject to our modeling assumptions.

1.2.2 Measurement models

Small rovers which move slowly operate in a quasistatic manner. The motion of the rover is not significantly influenced by dynamics. There are typically a few layers of software and hardware between the high level action plan, low level control, motor amplifiers, and the actual motion of motors, transmissions, and wheels. Sensors that directly measure the motion of the wheels return more reliable information about the motion of the robot than directives reported by higher level software such as planners, executive instructions or even low level control inputs. For this reason, the vehicle model used in this thesis is a measurement model, rather than a predictive dynamic model. The vehicle is assumed to have some means of measuring odometry and returning an estimate of the motion on the plane since the last odometry measurement was reported.

The odometry measurement is

$$\mathbf{d_i} = \mathbf{f}(\mathbf{m_i}, \mathbf{m_{i-1}}) + \mathbf{w_i} \tag{1.4}$$

where the components of the measurement are

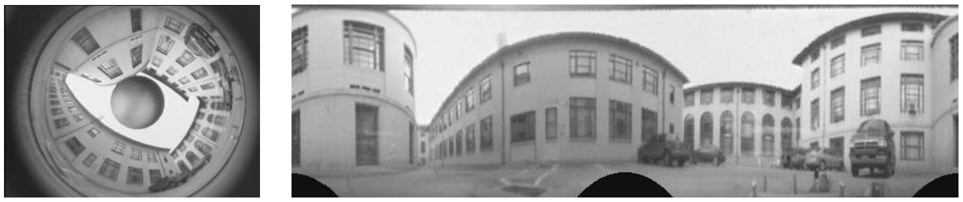
$$\mathbf{d_i} = \begin{pmatrix} \Delta m_x \\ \Delta m_y \\ \Delta m_\theta \end{pmatrix} \tag{1.5}$$

The noise term w_i is assumed to be zero mean with variance $\mathbf{Q_i}$, or

$$\mathbf{w_i} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q_i}) \tag{1.6}$$

By combining (1.4) and (1.6), we can write that the odometry measurement is distributed as a Gaussian with mean given by the odometry measurement equation and variance $\mathbf{Q_i}$, or

$$\mathbf{d_i} \sim \mathcal{N}(\mathbf{f}(\mathbf{m_{i-1}}, \mathbf{m_i}), \mathbf{Q_i}) \tag{1.7}$$



Raw image Dewarped image

Figure 1.2: Image from an omnidirectional camera

This motion information is enough for the robot to localize using dead reckoning, or path integration. However, path integration is known to produce poor pose estimates since the noise is integrated over time, producing error which grows unbounded.

A typical camera projects a 3-D scene onto a 2-D image plane. The depth to each point in the scene is lost. With an omnidirectional camera, the field of view that is projected onto the image plane consists of a full 360^o surrounding the camera. These camera systems are typically built from a normal CCD camera which looks upward at a curved mirror[63].

The omnidirectional camera projects the scene onto an annular region on the image plane, as shown on the left side of Figure 1.2. The elevation of a scene point with respect to the camera determines the distance from center of the image plane to the feature location on the image plane. The azimuth angle toward any scene point determines the angle around the annulus where the point appears on the image plane. This property makes the omnidirectional camera a viable way of measuring bearings toward features. The right side of Figure 1.2 shows a *dewarped* image, where the projection has been rectified to show the image such that the elevation angle determines the pixel row and the azimuth determines the pixel column in the rectified image. These images are more intuitive than the raw camera image. The distance from the left edge of the dewarped image is the bearing toward the feature.

In relation to the model of image formation for typical perspective projection cameras, the measurement model can also be thought to project a feature point in the world onto a unit circle (unit sphere in three dimensions) centered at the sensor. This model is depicted in Figure 1.3. The projection is nonlinear.

The bearing measurement equation

$$b_k = g(\mathbf{m_{i(k)}}, \mathbf{s_{j(k)}}) + v_k \tag{1.8}$$

models the bearing from the rover at pose $\mathbf{m_{i(k)}}$ to a landmark at position $\mathbf{s_{j(k)}}$, where $i(k)$ and $j(k)$ indicate which pose and landmark correspond to bearing k. The bearing measurement noise v_k is modeled as i.i.d. Gaussian with mean zero and covariance $\mathbf{R_k}$, or

$$v_k \sim \mathcal{N}(0, R_k) \tag{1.9}$$

Using (1.8) and (1.9), we can write that the bearing measurement is distributed as a Gaussian random variable with mean given by the measurement

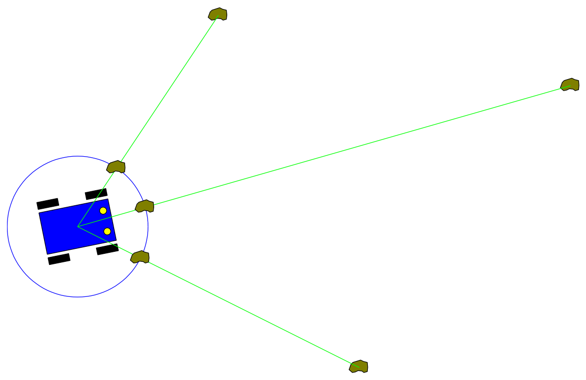


Figure 1.3: Bearing measurement model.

function and variance given by R_k,

$$b_k \sim \mathcal{N}(g(\mathbf{m_i}, \mathbf{s_j}), R_k) \tag{1.10}$$

The actual form of the measurement function $g()$ is

$$g(\mathbf{m_i}, \mathbf{s_j}) = \text{atan2}(s_{yj} - m_{yi}, s_{xj} - m_{xi}) - m_{\theta i} \tag{1.11}$$

Similar to the combination of all parameters describing the state, we define the measurement vector

$$\mathbf{z} = \{\mathbf{d_1}, \mathbf{d_2}, \ldots, \mathbf{d_i}, b_1, b_2, \ldots, b_k\}. \tag{1.12}$$

to be the collection of all odometry and bearing measurements. The generating function for all measurements as a function of all parameters is

$$\mathbf{z} = \mathbf{h}(\mathbf{x}) + \mathbf{v}, \tag{1.13}$$

where $\mathbf{v} \sim \mathcal{N}(0, \mathbf{R})$ is a normally distributed random variable. Following the assumption of i.i.d. noise for the individual measurements described above, $\mathbf{R}$ is (block) diagonal. The generating function for a single odometry or bearing measurement will be denoted $\mathbf{z_k} = h_k(\mathbf{x}) + v_k$ where $\mathbf{z_k}$ can be either type of measurement.

Using the notation described above, we can express the bearing only SLAM problem as estimating the parameters describing the rover trajectory and feature map using the measurements. In terms of a maximum a posteriori estimate, we seek

$$\mathbf{x}^* = \text{argmax} P(\mathbf{x}|\mathbf{z}) \tag{1.14}$$

The remainder of this thesis will focus on how to achieve or approximate this maximum a posterior estimate for the robot motion and feature map given measurements of robot odometry and bearings toward features.

1.3 The nature of bearings-only SLAM

There are a number of ways in which we can investigate the nature of the bearings only SLAM problem. In [30] the authors investigate the nature of the bearing-range SLAM problem by looking carefully at the behavior of a Kalman filter solution to the problem. The trouble with this approach is that there may be some confusion between the nature of the underlying problem, and the nature of the solution being applied.

A complete SLAM system must deal with many sources of uncertainty. Measurement errors in odometry and bearings will occur, and may loosely fall into errors which are captured by the system model, or *inliers*, and those which do not fall into the system model, or *outliers*. Outliers present a problem because when the assumptions made in the model are violated the behavior of the algorithm may be quite different from what is expected. For this reason, methods which are robust to outlier measurements are desired.

Furthermore, a complete SLAM system must deal with uncertainty in data association. Features must be detected and recognized as the same in order to use multiple measurements to refine estimates of feature locations and use the map to navigate. If correspondence is not known, then techniques such as multiple hypothesis tracking[74], RANSAC[35], or EM [26] might be used along with the procedures outlined below.

1.3.1 SLAM as parameter estimation

The use of odometry for pose estimation in the absence of any other information is one example of *deduced reckoning*, also referred to as dead reckoning or ded reckoning. Dead reckoning is known to have many problems, the most significant of which is that the errors in incremental motion estimation accumulate and position error grows without bound. That is, since the motion measurements are essentially integrated or summed over time,

$$\mathbf{m_i} = \sum_{k=0}^{i} \mathbf{d_k} \tag{1.15}$$

Since each motion estimate $\mathbf{d_k}$ has some error, the final pose estimate is affected by all of the errors on all of the incremental motion estimates. The result is that the error on the robot position grows unbounded. Depending on the magnitude of the errors the pose estimate may become useless rapidly. Systematic errors such as inaccurate estimate of wheel diameter, wheelbase, tire friction, etc. contribute as well as random effects such as wheel slippage, uneven or changing surfaces, or external forces on the robot. Nonetheless, dead reckoning is a well posed problem. Given an initial robot pose and a set of estimated motions, dead reckoning can compute an answer and an estimate of the quality of the answer.

If we only use the bearing sensor without information from odometry, then we can look to Structure from Motion (SFM) to gain some insight into what is possible. There is a counting argument often used in SFM to determine when

	# poses						
# features	1	2	3	4	5	6	7
1	N	N	N	N	N	N	N
2	N	N	N	N	N	N	N
3	N	N	N	N	N	N	N
4	N	N	N	Y	Y	Y	Y
5	N	N	Y	Y	Y	Y	Y
6	N	N	Y	Y	Y	Y	Y
7	N	N	Y	Y	Y	Y	Y

Table 1.1: Cases for which a solution exists in 2-D SFM.

solutions will exist for multiframe SFM. If there are N features and M robot poses from which bearings are measured, then there are $2N + 3M$ unknowns in the parameter estimation, 2 unknowns for the position of each landmark on the (x, y) plane and 3 unknowns (x, y, θ) for the pose of the robot at the time each measurement was taken. Let us assume for simplicity that all features of interest are visible in all frames, so that there are NM measurements. However, in the bearings only SLAM problem, as in SFM, there are a few *gauge freedoms*, or inherent ambiguities, particularly the coordinate frame and the scale factor. Any solution $\{\mathbf{m}, \mathbf{s}\}$ to the problem is also valid if a similarity transform is applied. A similarity transform in $2D$ has four parameters, translation in x or y (2 parameters), rotation (1), and scale (1). This means that to have a unique solution to the bearing-only SLAM problem, we need to include 4 constraints. A solution exists when the total number of constraints and measurements is at least as many as the number of unknown model parameters, or

$$4 + NM > 3M + 2N \tag{1.16}$$

which after some algebra becomes

$$(M - 2)(N - 3) > 2 \tag{1.17}$$

Looking at equation (1.17) we can see that without odometry, we cannot find a unique solution unless there are at least three robot poses and at least four landmarks, since otherwise either the left or right factor on the right side of the inequality is nonpositive. Figure 1.4 shows why the bearing measurements made from two consecutive robot poses do not help to disambiguate the robot motion. Even if the first robot pose is known to be at the origin, there are an infinite number of solutions to the second robot pose that are consistent with the bearing measurements: as long as the rays corresponding to the measurements intersect somewhere, the feature position can be triangulated.

For the special case of four features and three poses, the solution is still underconstrained. Table 1.1 enumerates the solvable cases for small N and M, and for large N and M the solution is always well constrained.

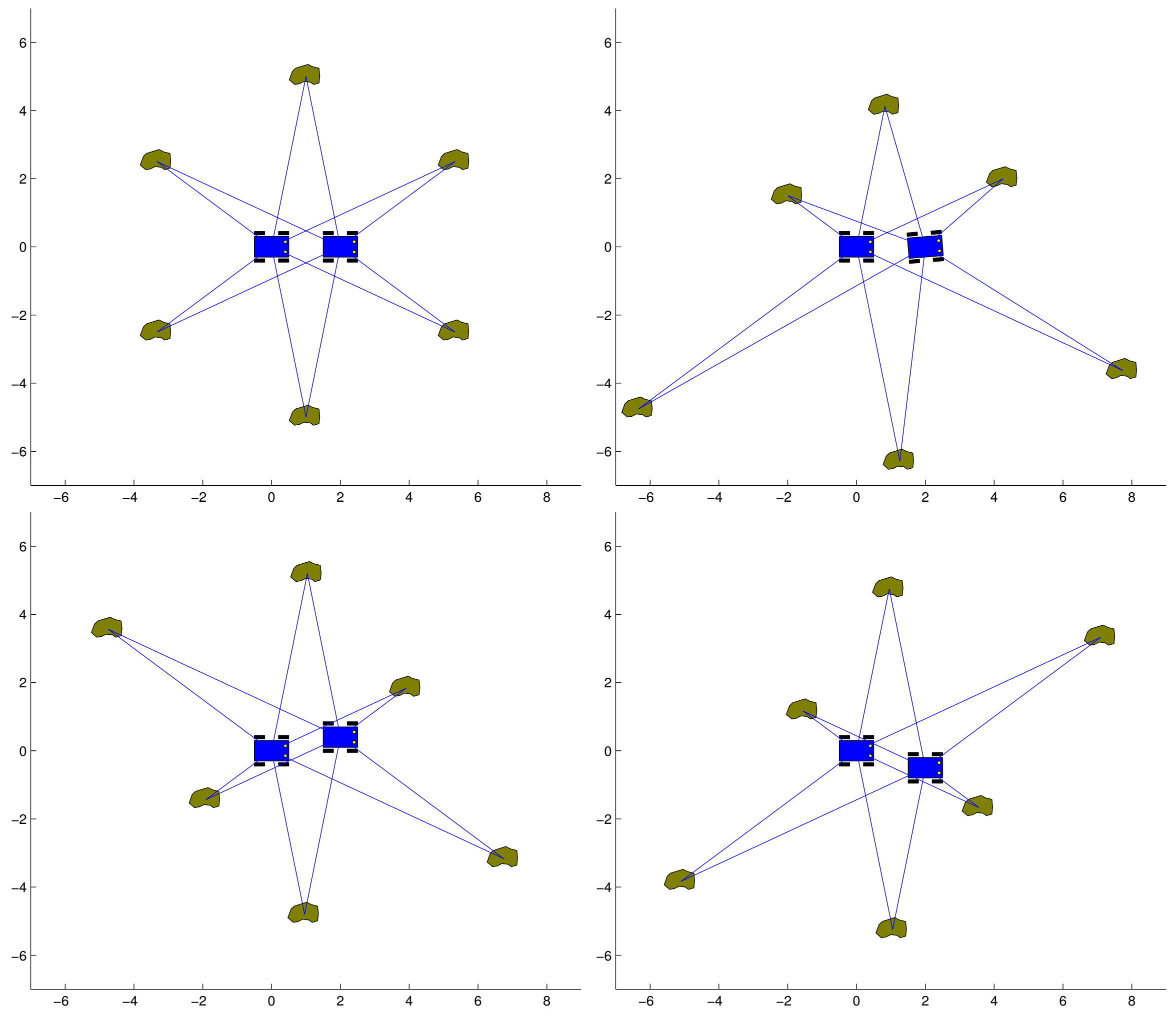


Figure 1.4: Ambiguity of two frame SLAM problem without odometry

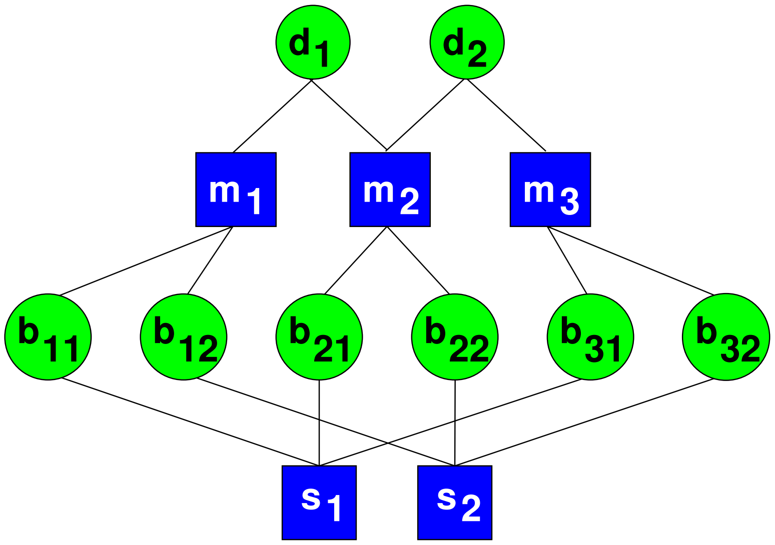


Figure 1.5: Conditional independence between measurements and states.

Using odometry alone, the problem is properly constrained. In fact, with redundant motion information, say with wheel odometry plus INS, the problem is overconstrained. But the fact that the problem is well posed does not mean that good estimates are available. It is well established that dead reckoning error grows unbounded, and that typically dead reckoning does not produce useful position estimates over long periods of time. But because the dead reckoning provides enough information to estimate pose, any bearing information at all will help to correct odometry errors. This has been demonstrated in the past, for example, by using fixation on a single feature in order to use a single bearing to improve navigation[1, 2].

1.3.2 Conditional dependencies

As a Bayes inference problem, the SLAM problem can be investigated in terms of the dependencies of measurements on particular subspaces of the state space. A graphical model is a convenient way of visualizing these relationships. Figure 1.5 shows one such visualization. Measurements are z (bearing) and $\mathbf{d}$ (odometry), represented by the circles, and the state space is made up of robot poses $\mathbf{m}$ and landmark positions $\mathbf{s}$, represented by squares. An odometry measurement depends only on two consecutive poses of the robot. The odometry system measures the motion of the robot from $\mathbf{m}_{i-1}$ to $\mathbf{m}_i$, and only that motion. Measurement $\mathbf{d}_i$ is conditionally independent of the rest of the state space and the rest of the measurements given $\mathbf{m}_{i-1}$ and $\mathbf{m}_i$. Likewise, the bearing sensor measures the direction toward a landmark from the current robot pose. Measurement z_{ij} is conditionally independent of the rest of the state space and

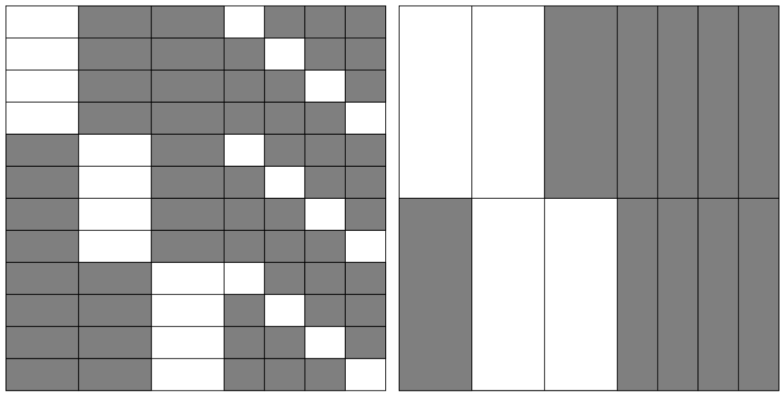


Figure 1.6: Sparseness of Jacobian induced by conditional independences.

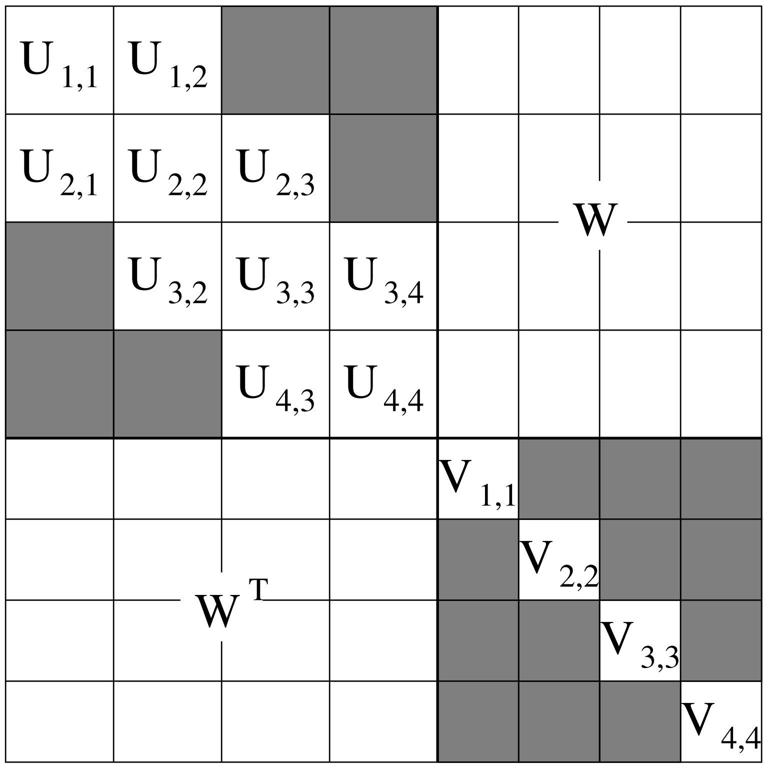


Figure 1.7: Sparseness of Hessian induced by conditional independences.

the rest of the measurements given $\mathbf{m}_i$ and $\mathbf{s_j}$. These conditional independences lead to the sparse nature of the estimation problem and therefore to fast algorithms for finding its solution.

1.3.3 Gauge freedoms

Similar to Structure from Motion, the SLAM problem often has the freedom to choose a coordinate system in which to estimate the position of the vehicle and the landmarks. Any choice of coordinate system is equally valid if there are no external references such as compass to disambiguate orientation and GPS to disambiguate absolute position. Typically this gauge freedom is removed by arbitrarily setting the initial position and orientation of the vehicle to be the origin of the coordinate frame in which the estimation takes place. However, due to biases or errors, the algorithm may still end up with a map which is close to the true solution but transformed to another coordinate frame.

Unlike the Structure from Motion problem, the SLAM problem should not have a scale ambiguity. Without motion information in SFM, camera motions and scene geometry can be arbitrarily scaled and still be consistent with the image measurements. In most SLAM work, the range measurements provided by a sensor such as sonar, radar, or laser disambiguate the scale, even when there is no motion information. In bearing-only SLAM, the scale ambiguity can only be resolved by vehicle odometry. This means that systematic errors in odometry may be unrecoverable if motions are always over- or under-estimated by the vehicle. This is equivalent to a change of units, e.g. from centimeters to inches. The resulting map should be consistent with future observations and may still be useful from the standpoint of navigation and path planning, but may contain an unrecoverable error in scale.

1.3.4 The linearization issue

Linearization is essential to the nonlinear least squares framework. Local derivatives are required for nonlinear optimization techniques, and recursive techniques for filtering, smoothing and prediction require linearization of the nonlinear process and measurement models in order to propagate the statistics of probability distributions. The way in which this linearization is done can have a large impact on the performance of the algorithms.

One of the principal concerns in Kalman filtering is that of filter *consistency.* If the filter estimates a covariance matrix that is too small, (or equivalently an inverse covariance matrix that is too big) the filter will begin to "trust" its prior estimate too much, and will not incorporate enough of the information that is coming from new observations. The result can be strong bias or even complete failure of the filter in the form of divergence.

For the bearing-only SLAM problem in two dimensions, we can get an understanding of the errors which occur in estimation by looking at how the linearization is influenced by errors in the state estimate. The measurement function in the bearing-only SLAM problem in the plane can be written as

$$h(\mathbf{m}, \mathbf{s}) = \operatorname{atan2}(s_y - m_y, s_x - m_x) - m_\theta \tag{1.18}$$

where the subscript m denotes motion (robot) and the subscript s denotes structure (landmarks). The measurement model is a Gaussian in measurement space

$$p(z) = k \exp\{-(z - h(\mathbf{x}))^T R^{-1} (z - h(\mathbf{x}))\} \tag{1.19}$$

or

$$z \sim \mathcal{N}(h(\mathbf{x}), R) \tag{1.20}$$

With one measurement of a feature, the Bayes posterior is

$$p(\mathbf{x}|z) = \frac{p(z|\mathbf{x})p(\mathbf{x})}{p(z)} \tag{1.21}$$

The prior probability for a given measurement $p(z)$ remains constant. If there is no prior information about where the feature is, then we can model $p(\mathbf{x})$ as uniform over state space, so that

$$p(\mathbf{x}|z) \propto p(z|\mathbf{x}) \tag{1.22}$$

Figure 1.8 shows the posterior for the landmark position after one measurement. The true posterior is a wedge-shaped region with a Gaussian profile in the direction orthogonal to the bearing from the robot to the feature, and is uniform along the bearing measurement direction since the measurement tells us nothing about the range to the feature. The measurement likelihood is Gaussian in measurement space,

$$p(z|\mathbf{x}) \propto e^{(z-h(\mathbf{x}))R^{-1}(z-h(\mathbf{x}))} \tag{1.23}$$

but the posterior must be modeled as a Gaussian in state space by linearizing the measurement function. The linearization of the measurement equation typically is the first order Taylor expansion,

$$h(\mathbf{x}) = h(\mathbf{x}_0) + H(\mathbf{x} - \mathbf{x}_0) \tag{1.24}$$

where $H = \nabla h$. The posterior Gaussian in state space resulting from this linearization is

$$p(\mathbf{x}|z) \propto e^{(z-h(\mathbf{x}_0)-H(\mathbf{x}-\mathbf{x}_0))R^{-1}(z-h(\mathbf{x}_0)-H(\mathbf{x}-\mathbf{x}_0))} \tag{1.25}$$

In order to understand the influence of the derivative of $h()$ on the posterior state estimate, we can look at the terms of the analytic Jacobian,

$$\nabla h = \begin{pmatrix} \frac{s_y - m_y}{r^2} & \frac{s_x - m_x}{r^2} & -1 & \frac{m_y - s_y}{r^2} & \frac{m_x - s_x}{r^2} \end{pmatrix} \tag{1.26}$$

where r is the distance between the robot and the landmark. The derivatives of the measurement with respect to the robot position and landmark position scale

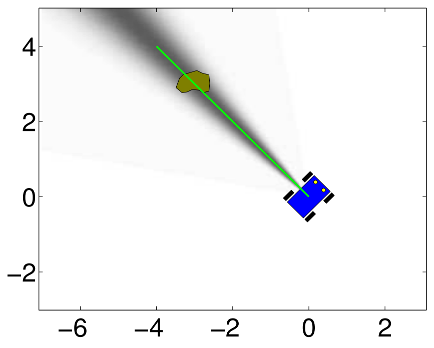


Figure 1.8: True posterior after one measurement

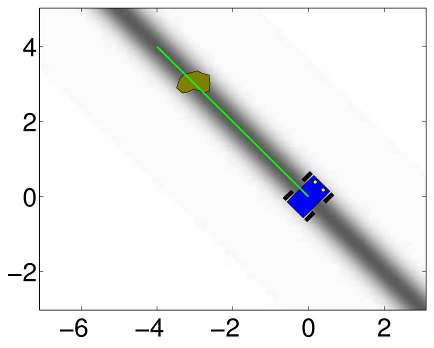


Figure 1.9: Posterior given by linearization at correct location

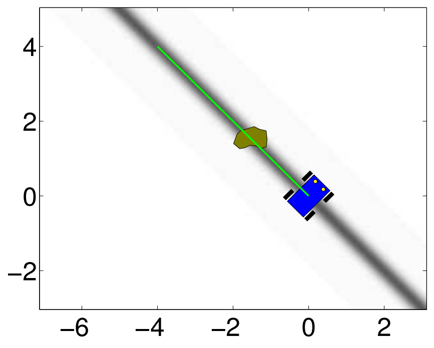


Figure 1.10: Posterior given by linearization at a closer point

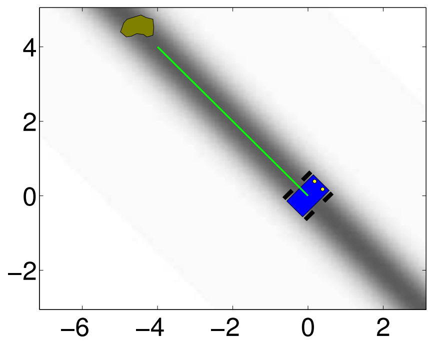


Figure 1.11: Posterior given by linearization at a farther point

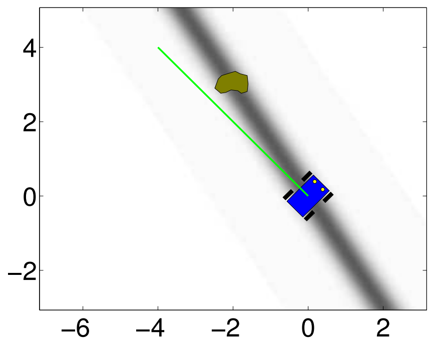


Figure 1.12: Posterior given by linearization at translated point

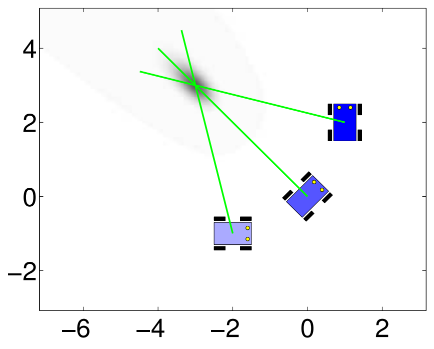


Figure 1.13: Posterior after three measurements

with the inverse of the distance between the landmark and robot. This means that when the landmark is closer to the robot, the certainty with which the filter can estimate the relative positions of the robot and feature will increase. For far away features the certainty in robot position and feature position decreases. The derivative with respect to the robot orientation does not change, so the certainty of the robot orientation is the same whether the features are near or far.

The linearization in the filter is performed at the current maximum likelihood or maximum a posteriori estimate of the state given the measurements. Figure 1.9 shows the posterior distribution approximated by a Gaussian in state space. The width of the Gaussian is the width of the wedge in Figure 1.8 at the current estimated position of the landmark. Note that because the posterior in state space merely represents a linear projection, the uncertainty region now extends infinitely far along the bearing direction, and on both sides of the robot. In other words, positions which would have the wrong bearing measurement by 180 deg in measurement space have high probability after linearization.

If the filter currently has an estimate for the feature position which is closer to the robot than the true landmark position, then the filter will compute a derivative which is too large and hence a posterior covariance which is too small. This effect can be seen in Figure 1.10. If the filter has an estimate for the landmark position which is farther away than the true position, the derivative will be too small and the covariance will be overestimated as shown in Figure 1.11. This is probably less of a concern when considering filter consistency, but down-weighting prior information can have negative impact as well. Finally, linearization computed at a point which is translated away from the ray corresponding to the bearing measurement will cause a covariance region with the wrong orientation as shown in Figure 1.12.

This analysis of state estimate errors leads to two desiderata for an algorithm for recursive bearing-only SLAM.

- The algorithm should compute smoothed estimates of the state in order to refine the estimated landmark and robot positions before linearizing.
- The algorithm should also use a linearization which is not simply computed at a single point.

The first of these points relates to where the linearization occurs. Refined estimates should be closer to the estimate which would be achieved by a full nonlinear estimator, and therefore the linearizations should be more accurate with respect to the full Bayes posterior. The second relates to how they are computed. A problem with the Jacobian approach is that the linearization takes place at the point corresponding to the maximum a posterior estimate, and is valid only close to that point. If we instead compute a linearization which is valid over the region where the posterior probability is large then we can expect the linearization to be valid when the point estimate changes. This thesis will explain a method for computing posterior statistics using a deterministic sampling procedure which outperforms analytic Jacobians.

1.4 Contributions

This thesis discusses the nature of the bearing-only SLAM problem and the relative advantages and limitations inherent to many existing methods applied to the problem. What is revealed by applying these methods motivates the development of a new method for solving the problem. The new method is based largely on two recent algorithms from the SFM and SLAM literature, and incorporates ideas from Monte Carlo techniques, numerical integration, and derivative-free state estimation.

The bearing only problem is interesting for many reasons. The motion and bearing measurement models are nonlinear. The bearing measurement is a projection, which loses information about the relative position of the feature since the range is not known. The full state can be of very high dimension since there are several parameters for each robot pose and for each feature in the map. The robot moves but map features do not, making the state space partly dynamic and partly static. Features may not be visible from all locations, and measurements may be made which grossly diverge from the true value or do not even correspond to actual features. This thesis presents a recursive technique which can deal with the nonlinearities, degeneracies, high dimensionality, and outlying measurements inherent to the bearing only SLAM problem. In particular, this thesis will show that principles from robust estimation, recursive nonlinear smoothing, statistical linearization, and systematic sampling can be combined to produce a recursive filter for SLAM that outperforms existing methods in terms of robustness and accuracy, with little increase in computational complexity.

1.5 Organization

The first contribution of the thesis is to investigate the application of several existing techniques to the problem of bearings-only SLAM to determine how well these methods work, where they fall short, and what can be learned about the difficulties in bearings-only SLAM as a result. Chapter 2 presents a brief overview of related work, primarily covering methods from the SLAM and SFM literature.

Chapter 3 presents an overview of batch estimation techniques, including the application of projective factorization (Section 3.1), nonlinear bundle adjustment (Section 3.2), and iteratively reweighted least squares for robust estimation (Section 3.3). The bundle adjustment and IRLS implementations are fairly straightforward applications of popular techniques, largely useful as a "gold standard" against which to compare other methods. The specialization of projective factorization to 2D bearings-only SLAM is a minor contribution of this thesis and provides some useful insights, but practicality of the technique for SLAM is limited since the framework does not make use of odometry measurements.

Chapter 4 discusses hierarchical methods, including strategies for filtering invariants (Section 4.1), and for divide-and-conquer style submap estimation

(Section 4.2). The invariant filtering technique presented is a nonlinear extension to an earlier linear technique which required a fixed heading reference[22].

Chapter 5 presents an overview and analysis of recursive estimation techniques applied to bearings-only SLAM, in particular the Kalman filter (Section 5.2), the nonparametric Unscented and Divided Difference ilters (Section 5.3), and fixed lag smoothing (Section 5.4). These sections provide much of the components and motivations for the main contribution of the thesis in Chapter 6.

Chapter 6 is the main contribution in the thesis. Chapter 6 discusses in detail how to apply smoothing (Section 6.1) and statistical linearization (Section 6.2) to the problem. A major problem with applying statistical linearization is in the computation of expectation values over the filter posterior. This is addressed in Section 6.3 which discusses numerical integration techniques and develops a method for efficiently and accurately computing the integrals in Section 6.2. The other major difficulty with computing the expectations in Section 6.2 is the curse of dimensionality, which is addressed in Section 6.4 by analyzing the structure of the bearings-only SLAM problem to exploit conditional independences, gauge freedoms, and the hybrid linear/nonlinear nature of the problem in order to minimize the number of dimensions over which integrals must be computed. The result is a novel algorithm which combines the advantages of several previous methods in a way that was not formerly feasible. The algorithm is presented in detail in Section 6.5. Section 6.6 presents a further addition of robust methods, and Section 6.7 presents a further addition of computational efficiency by maintaining square root matrices within the filter rather than full matrices.

Chapter 7 presents experiments designed to demonstrate the performance of the new method. A comparison to the performance of an EKF are presented in Section 7.1.1. A comparison of the performance of point derivatives vs. statistical linearization is presented in Section 7.1.2 which covers the improved expected map errors, and in Section 7.1.3 which covers the improvement in consistency of the filter. Section 7.2 presents experiments with real data from an ATRV Jr. operating in an unstructured environment at a test facility at NASA Ames Research Center, and a Pioneer DX2 robot operating on campus at CMU.

Chapter 8 presents a discussion of the method. Section 8.1 discusses qualitative and quantitative metrics on the performance of the algorithm, including consistency, convergence, complexity, optimality, and reliability. Section 8.2 discusses its relationship to many popular existing methods. Section 8.3 identifies some directions for future work.

Chapter 2

Related work

There are two areas of research which are most directly related to the bearings only simultaneous localization and mapping (SLAM) problem. The first is of course the community of researchers investigating SLAM, also known as Stochastic Mapping, Concurrent Localization and Mapping, etc. The second is the body of literature concerned with the Computer Vision problem referred to as Structure from Motion (SFM) or Multi-Frame Structure From Motion (MFSFM).

The goal of the bearing only SLAM problem is to reconstruct a map of the environment and an estimate of robot motion given bearing measurements and motion measurements. In Structure from Motion, a camera provides feature measurements. The camera measurement is a projection of the environment. There is typically no motion measurement in SFM.

By contrast, most SLAM work makes use of both motion measurements from INS, GPS, odometry, etc. and range-bearing sensors such as laser rangefinders, radar, or sonar. The feature measurements are not a projection of the environment, but rather a rigid transformation.

Bearing only SLAM contains a feature measurement model which is a projection of the environment, which is similar to SFM, and contains a motion measurement model, as does the original SLAM problem. For this reason, literature from both areas is relevant to this thesis.

In some ways the problem of range scan alignment is related to SFM and SLAM as well. Range scan alignment seeks to register data from a sensor which measures a full 3D model of an environment or an object, and often without any prior information on the motion of the sensor or the object between scans. The problem is quite different from bearing only SLAM, however, so we will not discuss literature from range scan alignment but merely point out its relationship to these other related problems.

A diagram showing the relationship between structure from motion, simultaneous localization and mapping, range scan alignment, and bearing-only SLAM is shown in Figure 2.1.

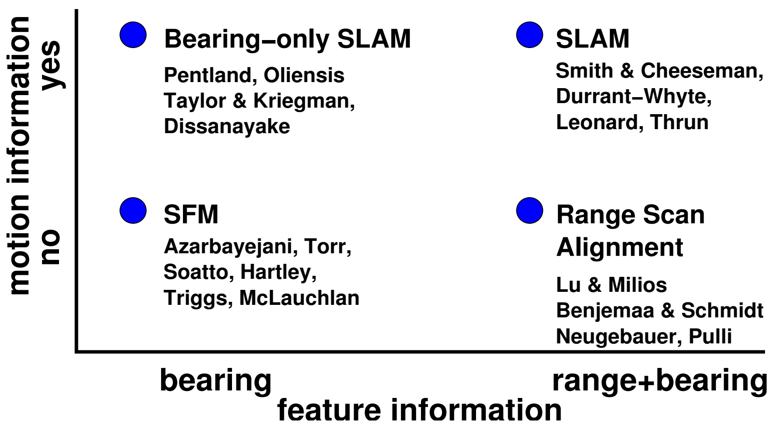


Figure 2.1: Problems related to bearing only SLAM

2.1 Simultaneous localization and mapping

The SLAM problem has been of interest for just over a decade. Kalman filters are used in a large majority of the methods reported[79, 62, 47, 30, 34], although other recursive approaches include estimating invariants[22], unscented filtering[44], covariance intersection[92], and submap estimation methods[34]. Batch approaches have also been proposed, including range scan alignment[50], expectation-maximization[10], and nonlinear least squares[84].

2.1.1 Stochastic mapping

The seminal paper by Smith and Cheeseman on uncertainty representation in robotics presented the *stochastic map* algorithm, a Kalman filtering solution to SLAM[80]. Moutarlier and Chatilla presented the first experiments using a real robot[62], and also presented two different ways of representing relationships between objects, the first they called *relation* representation and the second *location* representation[62]. In the relation representation, each object has its own reference frame and is related to other objects by uncertain transformations between their reference frames. In location representation, each object uses the same global reference frame, and the object's location in the global coordinate frame is the parameter of interest.

A decade of SLAM research largely adopted the location representation, and typically used recursive estimation procedures, almost exclusively Kalman Filters or methods directly related to Kalman Filters, essentially the stochastic mapping algorithm of Smith and Cheeseman[79]. Approaches based on these methods are common and a good review can be found in [34]. These methods usually estimate the entire map and the most current vehicle pose[30, 47]. The implementations frequently assume that odometry is very accurate and that bearing and range are measurable.

2.1.2 Kalman filter extensions

There have been two important extensions to Kalman filtering recently in the SLAM community. The first is Covariance Intersection[92], which deals with consistent estimation in the presence of unknown correlations. Covariance Intersection is a pessimistic estimator whereas the Kalman filter, by assuming that all measurements are completely uncorrelated, can be overly optimistic. This optimism can cause divergence problems.

The second improvement to Kalman filtering in SLAM is Unscented Filtering, in which the mean and covariance of the filter estimate are computed using a deterministic sampling scheme[44]. Samples are drawn with the exact first through fourth moments and passed through the nonlinear model. The posterior moments of the samples are computed and used in place of the mean and covariance computed using the Kalman filter equations. The Divided Difference Filter[68] uses a similar deterministic sampling scheme for interpolation with very similar results. This deterministic sampling will be discussed later in terms of Gaussian quadrature, and generalized to computing approximate sufficient statistics in a slightly broader recursive filtering context.

2.1.3 Estimating invariants

Recently, there have been several methods which attempt to use an invariant relationship approach to SLAM. Csorba's Relative Filter [23] recursively updates minimalistic relationships between landmark features in a Kalman filtering framework. Each pairwise (or n-tuple-wise) relationship is estimated independently. The advantage in relative filtering is that the measurements which influence the estimate of a relationship between two features might not directly influence the estimate of another relationship. The parameterization gains by decoupling the state space, causing the state covariance to become block diagonal and offering fast updates to the invariant features. The disadvantage is that in order to localize, navigate, or plan paths using the map, a step is required which converts the information about relationships back to a consistent global map.

The approach was extended to enforce global map consistency by Newman with the Geometric Projection Filter (GPF)[66]. The GPF provides a means by which to produce a geometrically consistent map from the relative features in Csorba's filter by solving a set of simultaneous linear constraint equations to project the relative map states onto an absolute coordinate system. The relative filter and GPF assumed that absolute heading is measurable (e.g. by a compass) and that bearing and range to each target is measurable, in order to keep the problem linear. The method was later extended to nonlinear estimation and bearing-only measurements to loosen the requirement on accurately known heading[25].

2.1.4 Submap methods

Other work decouples the problem at a larger scale of local submaps, using a Kalman filter approach in a neighborhood but treating neighborhoods as independent and uncorrelated. The covariance matrix is again block diagonal, but the blocks correspond to many landmark features. Inversion becomes simpler and the EKF update can be done using only the states relevant to a local map. Leonard and Feder's Decoupled Stochastic Mapping (DSM) algorithm[49] tessellates the environment in this way with loosely coupled local maps. Between local maps the relationship is approximate and covariances are not considered. The computational complexity of the *stochastic map* is incurred only within each local map.

The Expectation-Maximization algorithm has been used to register and combine local maps built with sonar and dead reckoning[10]. In this approach, it is assumed that odometry is accurate over short distances and mapping is done with approximate pose information in a local coordinate frame. EM is used to register and combine local maps to build a global map.

2.1.5 Batch methods

Some methods estimate the whole history of vehicle poses and the map, which causes a linear growth in the dimension of the state vector but allows smoothing of the trajectory estimate at each step [50, 84].

SLAM methods that do not use the Kalman filtering framework often use a least squares approach. Lu and Milios[50] have developed a SLAM method which is very similar to the Iterated Closest Point (ICP)[14, 6] algorithm, and Taylor et al.'s nonlinear least squares optimization method[84] is similar to bundle adjustment, which will be discussed below.

2.1.6 Image registration

Image correlation[39] has also been explored for the SLAM task. Leveraging on a large development effort in fast image registration, Konolige built a system which converts sonar scans into local grid based maps which are represented as "images" and uses fast image registration techniques to combine them into a large composite grid. The technique has been applied to indoor environments. As the robot moves and measures sonar scans, each local scan is registered to the current global map. The robot keeps a measurement history and if the robot travels in a loop and detects loop closure, the measurements taken on that loop are all registered together to improve the estimate over the sequential registration.

2.1.7 Rao-Blackwellized particle filters

Based on our own experiments, a straightforward implementation of particle filters does not perform well for bearing only SLAM. The biggest problem is in initializing new features with one measurement, since the feature can lie anywhere along an infinite ray and the full posterior is represented using only samples.

Rather than sampling over the full joint posterior, the Rao-Blackwellized particle filter (RBPF) maintains samples for a marginal distribution over some part of the state space, then for each sample computes a parametric conditional distribution over the rest of the state space[13]. Recently, Montemerlo and Thrun[59] have worked with a RBPF for range-bearing SLAM. By choosing to marginalize and sample over robot pose and then estimate parametric densities for landmark positions, the landmarks can be mapped independently. The map updates are then trivial; the update can be done in $O(N)$ for N landmarks. The approach has not yet been extended to the bearing-only case.

Within a RBPF algorithm, some method is required for maintaining the parametric density estimates for the conditional probabilities. One common approach is to use a Kalman filter. The recursive estimation procedure described in this thesis outperforms Kalman filtering and can therefore be used in conjunction with a RBPF scheme.

2.1.8 Bearings only SLAM

Taylor and Kriegman published a nonlinear least squares approach to the bearing only SLAM problem in 1991[84]. The approach was a batch method, and used Gauss Newton to alternate between solving for an update to the trajectory parameters given the landmarks and solving for an update for the landmark parameters given the trajectory. The justification for this was that each of these updates was trivial to compute, taking only $O(M)$ for updating M robot poses and $O(N)$ to update N landmarks. However, although each update step has quadratic convergence within one step, the combination of updating pose given map and then map given pose ignores the relationship between the two and the overall algorithm has sublinear convergence. A recent review paper on bundle adjustment[90] details the limitations of algorithms like Taylor's, arguing for a good approximation to the Hessian of the full objective function in order to get fast overall convergence at the expense of more computationally expensive steps.

Another recent paper approaches bearings only SLAM as a Structure from Motion problem in the plane[97]. The algorithm uses nonlinear bundle adjustment to compute local maps, then combines local maps to produce a global map.

2.2 Structure from motion

The photogrammetry and computer vision literature contain a significant amount of work related to the structure from motion (SFM) problem, in which monocular images alone are used to reconstruct the scene and recover the camera motion. Much like the SLAM problem, this seems like an egg and chicken problem.

If we knew where the cameras were when the images were taken, then reconstructing the scene would be a matter of triangulation. If we knew the structure of the scene, then we could localize the camera again using triangulation techniques. However, in the general SFM problem there is no a priori knowledge of the scene and no a priori knowledge of the camera positions. Among the popular approaches are factorization [88], sequential multiframe geometric constraints [76], recursive filtering[8, 4], and nonlinear bundle adjustment[90].

It has been known to photogrammetrists for many decades that for a 3D scene and two views, as few as five corresponding points constrain the structure of the scene and the relative positions of the camera views up to a similitude, e.g. a Euclidean transformation and scale ambiguity. The Euclidean transformation is present because we can not know the absolute positions of the cameras, we can only recover their relative positions. It can be assumed that one of them was at a particular location in order to constrain the solution, i.e. define one camera to be at the origin. The scale ambiguity is introduced due to the fact that scaling the environment and the distance between camera positions by the same factor yields the same images, so from the images one can only hope to recover structure up to this scale factor. In practice, many SFM solutions simply assume that the translation between a pair of cameras is of unit length, or that the distance between two scene points is unity.

2.2.1 Factorization

A class of SFM solutions exist for orthographic projection which use factorization using SVD [88]. These methods are not applicable to perspective projection or bearings only sensing, however, since the projection is not a linear one. Factorization was later extended to work with weak perspective[70] and later full perspective projection[83] camera models. The recent algorithm from Mahamud and Hebert[53] exhibits improved performance over previous projective factorization techniques and can be extended to the bearing-only SLAM problem, albeit without odometry information. The advantage to projective factorization methods is that they do not rely on a good initial guess for convergence.

2.2.2 Recursive filtering

Recursive SFM methods have been reported, following the seminal work[8] in which an IEKF is used to recursively estimate motion and structure from images. The filter is initialized using a batch processing technique to interpret the first set of frames in the video sequence. The work was later extended to incorporate the estimation of focal length, using a novel camera parameterization to ensure that the problem remain well conditioned[4].

The general Structure From Motion problem assumes no a priori information about camera motion, which is overly conservative in our case since we may have some motion estimates from odometry, and we may have nonholonomic constraints or other knowledge which can help constrain the solution. Some recursive SFM work has incorporated inertial sensing as well[73].

2.2.3 Sequential linear constraints

There have also been 3D structure from motion algorithms which compute two-view structure and motion from pairs of images taken sequentially, then recursively filter the resulting structure estimates using an EKF or similar smoothing filter [69, 81]. These methods may be suitable for the current problem if three consecutive views are used. Faugeras published a paper on trilinear constraints for 1D cameras[32, 33], and Dellaert used the technique for SLAM[27].

2.2.4 Nonlinear optimization

Many SFM solutions have used nonlinear optimization techniques, most notably Levenberg-Marquardt, to handle cases where factorization is not applicable. Hartley explained the sparse nature of the SFM problem and detailed an algorithm which takes advantage of the structure of the problem to reduce computational complexity in the worst case[40]. A more recent review article discusses many aspects of nonlinear bundle adjustment, including a more generalized view of sparse matrix formulation of SFM as well as robust estimation and other improvements[90].

2.2.5 Robust estimation

Bundle adjustment is a least squares approach to estimation. As such it suffers from outliers. Measurements which do not conform well to a Gaussian error model can have drastic impact on the solution found by least squares. For this reason, robust methods have been sought for parameter estimation under the influence of outliers[42]. Many cost functions have been proposed, and a scheme exists for the use of weights in a least squares algorithm to achieve robustness. The iteratively reweighted least squares (IRLS) technique has been used for general parameter estimation problems[98] and tailored specifically to SFM[90], and is an important influence on the work presented in this thesis.

2.2.6 Recursive nonlinear smoothing

Nonlinear smoothing provides an important improvement over Kalman filtering for bearing-only SLAM. The Variable State Dimension filter (VSDF)[57, 55] combines the EKF with nonlinear Gauss-Newton, resulting in a smoothing algorithm. The advantage is that the filter contains multiple robot pose estimates in the filter, and treats the measurements from these states as nonlinear quantities until an accurate estimate is computed. This helps mitigate the bias problems that the EKF suffers from when linearizations are computed at the wrong point. The VSDF lays important groundwork for the method presented in this thesis.

2.2.7 Gauge constraints

Recently a lot of work by Morris[61] and McLauchlan[56] has been done which looks at the application of gauge constraints in the SFM problem. That work is applicable here as well. However, due to the mobile robot nature of the problem, the gauge constraint is typically applied by constraining the initial robot pose with some convention, e.g. that the robot starts at the origin with heading 0, and with no uncertainty in its pose since the pose is set by convention. The way in which this gauge constraint is enforced in practice depends on the algorithm being used. Batch methods can leave the gauge freedom in the problem until a solution is found, then project the rank deficient Hessian using techniques described by Morris[61] to the correct gauge. Recursive formulations must take the linearization and marginalization of the first pose as a special case in order to enforce the constraint.

2.2.8 Hierarchical methods

A hierarchical SFM approach is presented in [36] which builds submodels using trilinear constraints then merges submodels and recovers all scene and model parameters. However, in a model building or mapping application, the recovery of the camera parameters for every view may not be necessary and a considerable amount of computation could be eliminated.

Another hierarchical SFM approach is presented in [78] which computes local structure estimates and then projects them into virtual "key frames", eliminating the viewing parameters from the submodel estimation but also projecting the submodel into a small number of virtual views. It is not clear whether the key frames capture as much information about the reconstructed model as possible given the original data.

2.3 Range scan alignment

Many different approaches exist to the range scan alignment problem. The earliest methods sought to align two 3-D point clouds with known correspondences, and Horn's method[41] is perhaps best known. Later techniques registered multiple views either directly[5, 38] or by first computing pairwise alignments and then approximating the posterior over pairwise alignment to find a global registration[82, 72]. Gold et al. simultaneously found an alignment for point clouds and the correspondence between them[37], as does the ICP algorithm[14, 6]. Because the range scanner might measure distinct points on a continuous surface, Neugebauer incorporated an error metric which used the distance from a point to a surface, rather than the distance between corresponding points[64]. Lu and Milios used range scan alignment in the context of 2D SLAM[50].

2.4 Conclusions

Insights from bundle adjustment, Kalman filtering, robust statistics, variable state dimension filtering, and the unscented filter and divided difference filter are crucial to the development of the algorithm proposed in this thesis. Several of the methods briefly described above have been applied to the bearing only SLAM problem, and this dissertation will explain the adaptation of these methods and some of the insights and conclusions which resulted. The central theme of the thesis will then be the formulation of a robust nonlinear smoothing algorithm which uses deterministic sampling to compute posterior statistics.

Part II

Batch estimation

Chapter 3

Full batch estimation

Batch estimation uses all of the observations to solve for all of the unknown state variables at the same time. Batch estimation allows us to find a maximum *a posterior* (MAP) estimate

$$\begin{aligned} \mathbf{x}^* &= \operatorname{argmax}_{\mathbf{x}}\ p(\mathbf{x}|\mathbf{z}) \\ &= \operatorname{argmax}_{\mathbf{x}}\ p(\mathbf{z}|\mathbf{x})p(\mathbf{x}) \end{aligned} \tag{3.1}$$

where $p(\mathbf{z}|\mathbf{x})$ is the data likelihood and $p(\mathbf{x})$ is the prior on solutions. The goal of most recursive estimation algorithms is to approximate (3.1) as closely as possible. Therefore we will first consider the nature and solution of the SLAM problem in terms of a full batch style MAP estimation problem.

A projective factorization approach is outlined first. The approach is adapted from Structure from Motion and is directly applicable to the bearing measurements in the SLAM problem, but a formulation which takes odometry into account is not currently known. The advantage of projective factorization is that it is a linear algorithm, and does not require any initial guess for the parameters. The disadvantages are that each iteration is slow and it can take many iterations to converge, and that the solution is not guaranteed to be physically meaningful, i.e. the solution from projective factorization must be projected onto the set of feasible solutions, and may therefore not be optimal in a least squares sense.

An adaptation of nonlinear bundle adjustment from computer vision is outlined next. This approach is capable of finding optimal solutions in a least square sense, and provides a considerable amount of insight into the nature of Kalman type algorithms for recursive SLAM estimation. Iterations are fast and only a few are required for convergence, but the algorithm requires an initial guess at the parameters and can get stuck in bad local minima if the initial guess is not in the basin of attraction for the global optimum.

Finally some consideration is given to robust estimators, which are designed to perform reasonably in the presence of outliers in the observations.

3.1 Projective factorization

Factorization for structure from motion was first introduced by Tomasi and Kanade[88] for orthographic projection. It was later extended to work under weak perspective by Poelman [70] and more recently for perspective projection by Sturm [83]. An improved algorithm for projective factorization for perspective cameras by Mahamud[52, 53] is very simple and also exhibits good performance for 2D bearings only SLAM.

Orthographic projection models image formation as a completely linear process, and is only accurate for extreme focal length camera systems or extreme distances between camera and scene. Weak perspective approximates perspective projection by treating all points in the scene as though they are at the same depth, and therefore work reasonably well for scenes with little depth variation compared to average scene depth. Because a mobile robot may be moving in the scene that is being mapped, depth variations can be expected to be extreme, and the bearing measurement is a highly nonlinear model. Only the perspective projection methods are applicable to bearings only SLAM.

The measurement model may be written as

$$\mathbf{Z} = \mathbf{MS} \tag{3.2}$$

where $\mathbf{S}$ is a scene point and $\mathbf{M}$ is the robot motion parameterized by a projection matrix. That is, the entries in the projection matrix encode the rover trajectory.

For orthographic cameras, $\mathbf{Z}$ is an image point which is bilinear in the camera projections $\mathbf{M}$ and the scene $\mathbf{S}$. That is, the image formation in (3.2) is a linear model. The image is modeled as a simple multiplication without any scaling due to scene depth.

Under a perspective camera projection model, each 3-D point in the scene is mapped to a point on the image plane with some homogeneous coordinates $(u, v, 1)^T$. With a bearings only sensor model in a 2-D environment, the equivalent way to write the measurement as a homogeneous coordinate is $(\cot(z), 1)^T$, which would model a 1-D camera "plane". However, rather than use homogeneous coordinates where the last coordinate for each point is set to 1, we use the projective coordinate $(\cos(z), \sin(z),)^T$ which is much better behaved for points which would project to (or near) infinity on a 1-D camera, i.e. bearing measurements which are nearly parallel to the image plane.

Given the true bearing z_{ij} and range r_{ij} to each feature, the localization and mapping problem can be reduced to finding the rigid transformation which aligns sensor scans. With many range/bearing scans (and known data association) the measurements could be stacked into a measurement matrix $\mathbf{Z}$ as in

$$\mathbf{Z} = \mathbf{MS} = \begin{bmatrix} r_{11}\cos(z_{11}) & r_{12}\cos(z_{12}) & \cdots & r_{1N}\cos(z_{1N}) \\ r_{11}\sin(z_{11}) & r_{12}\sin(z_{12}) & \cdots & r_{1N}\sin(z_{1N}) \\ r_{21}\cos(z_{21}) & r_{22}\cos(z_{22}) & \cdots & r_{2N}\cos(z_{2N}) \\ r_{21}\sin(z_{21}) & r_{22}\sin(z_{22}) & \cdots & r_{2N}\sin(z_{2N}) \\ \vdots & \vdots & & \vdots \\ r_{M1}\cos(z_{M1}) & r_{M2}\cos(z_{M2}) & \cdots & r_{MN}\cos(z_{MN}) \\ r_{M1}\sin(z_{M1}) & r_{M2}\sin(z_{M2}) & \cdots & r_{MN}\sin(z_{MN}) \end{bmatrix} \tag{3.3}$$

where $\mathbf{M}$ is a vertical stack of projection matrices

$$\mathbf{M} = \begin{bmatrix} \cos(\theta_1) & \sin(\theta_1) & x_1 \\ -\sin(\theta_1) & \cos(\theta_1) & y_1 \\ \cos(\theta_2) & \sin(\theta_2) & x_2 \\ -\sin(\theta_2) & \cos(\theta_2) & y_2 \\ & \vdots & \\ \cos(\theta_N) & \sin(\theta_N) & x_N \\ -\sin(\theta_N) & \cos(\theta_N) & y_N \end{bmatrix} \tag{3.4}$$

which capture the set of robot poses from which the scans were taken, and

$$\mathbf{S} = \begin{bmatrix} u_1 & u_2 & & u_N \\ v_1 & v_2 & \cdots & v_N \\ 1 & 1 & & 1 \end{bmatrix} \tag{3.5}$$

is a column matrix of homogeneous coordinates for the landmarks. Since $\mathbf{S}$ is rank 3 and $\mathbf{M}$ is rank 3, the resulting matrix $\mathbf{Z}$ is rank 3. We can compute a factorization of $\mathbf{Z}$ using SVD in order to find a $\mathbf{M}$ and $\mathbf{S}$ that are consistent with the measurements. This is done in closed form, without any initial guess at structure or motion. In general, any valid factorization computed this way is a member of an equivalence class of projective solutions

$$\mathbf{Z} = \widehat{\mathbf{M}}\mathbf{Q}^{-1}\mathbf{Q}\widehat{\mathbf{S}} \tag{3.6}$$

where $\mathbf{Q}$ is a nonsingular projective transformation. For a useful Euclidean solution a $\mathbf{Q}$ must be found such that the final solution

$$\mathbf{M} = \widehat{\mathbf{M}}\mathbf{Q}^{-1} \tag{3.7}$$

satisfies orthonormality constraints[83, 89, 52] and then $\widehat{\mathbf{S}}$ is upgraded accordingly.

With real (noisy) data, SVD is used to find the best rank 3 factorization and under the right circumstances can yield the MLE estimate. This could be used to solve SLAM problems with bearing/range sensors, and related methods exist in 3D model registration and other related problems.

Without the range information the SVD approach cannot be applied directly because the measurements are not easily factorized. The basic idea in projective factorization is to approach factorization as a missing data problem where the bearing z_{ij} is known but the range r_{ij} is missing. We begin with a measurement matrix $\mathbf{Z}$ which consists of the sines and cosines of the bearing measurements z_{ij} similar to (3.3). A scaling parameter λ_{ij} is introduced (which in the projective factorization literature is referred to as *projective depth*) which is used to scale the bearing measurements yielding the scaled measurement matrix

$$\hat{\mathbf{Z}} = \begin{bmatrix} \lambda_{11}\cos(z_{11}) & \lambda_{12}\cos(z_{12}) & \cdots & \lambda_{1N}\cos(z_{1N}) \\ \lambda_{11}\sin(z_{11}) & \lambda_{12}\sin(z_{12}) & \cdots & \lambda_{1N}\sin(z_{1N}) \\ \lambda_{21}\cos(z_{21}) & \lambda_{22}\cos(z_{22}) & \cdots & \lambda_{2N}\cos(z_{2N}) \\ \lambda_{21}\sin(z_{21}) & \lambda_{22}\sin(z_{22}) & \cdots & \lambda_{2N}\sin(z_{2N}) \\ \vdots & \vdots & & \vdots \\ \lambda_{M1}\cos(z_{M1}) & \lambda_{M2}\cos(z_{M2}) & \cdots & \lambda_{MN}\cos(z_{MN}) \\ \lambda_{M1}\sin(z_{M1}) & \lambda_{M2}\sin(z_{M2}) & \cdots & \lambda_{MN}\sin(z_{MN}) \end{bmatrix} \tag{3.8}$$

An appropriate choice of λ_{ij} produces $\hat{\mathbf{Z}}$ which is approximately rank 3. However, arbitrary scaling of the columns and arbitrary scaling of pairs of rows will not affect the rank of $\hat{\mathbf{Z}}$, so a suitable choice of λ_{ij} will in general recover the structure and motion up to some projective ambiguity. Other constraints must be applied to recover Euclidean structure and motion.

The procedure begins by initializing a guess at the projective depths and an alternating minimization procedure is then used to produce estimates for $\mathbf{M}$ and $\mathbf{S}$ using a rank 3 decomposition of $\hat{\mathbf{Z}}$. $\mathbf{M}$ and $\mathbf{S}$ are then used to compute a new estimate for λ_{ij} by noticing that together $\mathbf{M}$ and $\mathbf{z}$ provide a set of linear constraints for $\mathbf{S}$. A generalized eigenvalue problem is solved to compute new values for λ_{ij}. Details can be found in [53].

Projective factorization has a few drawbacks as a practical solution to bearings only SLAM. The algorithm as described above uses only the bearing measurements. The motion as measured by odometry could be used to fix the scale of the reconstruction but there is no simple way to incorporate all of the information in the odometry (i.e. direction of motion, measured rotation) in order to produce better trajectory estimates. The computation of the SVD step is slow and many iterations are required since the convergence is sublinear. There is currently no incremental or recursive version. Finally the algorithm as described only deals with features that are visible from every robot pose, so the method would need to be extended to deal with occlusions, perhaps using something like the PCA with Missing Data (PCAMD) algorithm[77]. However, it should also be pointed out that as a linear method, projective factorization often has better convergence properties, i.e. gets stuck in local minima less frequently than nonlinear optimization methods might.

3.2 Nonlinear bundle adjustment

Bundle adjustment is the use of Levenberg-Marquardt to solve the multiframe Structure from Motion problem. It has been used in Photogrammetry for a few decades. The name "bundle adjustment" refers to an intuition for the optimization problem as the alignment of bundles of rays emanating from the camera centers and passing through the image planes and feature points. With little modification, Levenberg-Marquardt optimization can be used for the bearing only SLAM problem as well.

Bundle adjustment is a full nonlinear optimization. The algorithm computes first and second derivatives of an objective function at every step using all available observations, and updates its estimate of all model parameters. Since bundle adjustment optimizes all motion and structure parameters at every step, the state vector is the entire history of robot pose and the entire map, which can become quite large.

We begin with the Bayesian estimation expressed in (3.1) above, and model the measurements as conditionally independent

$$\begin{aligned} \mathbf{x}^* &= \operatorname{argmax}_{\mathbf{x}} p(\mathbf{x}) p(\mathbf{z}|\mathbf{x}) \\ &= \operatorname{argmax}_{\mathbf{x}} p(\mathbf{x}) \prod_i p(\mathbf{z_i}|\mathbf{x}) \end{aligned} \tag{3.9}$$

We assume a uniform prior over model parameters, so that $p(\mathbf{x})$ is constant and can be removed. The result is a maximum likelihood estimation problem. The observation model assumes that measurements are a function of the model parameters with some Gaussian noise,

$$p(\mathbf{d}, \mathbf{b}|\mathbf{x}) = k e^{-\frac{1}{2}(\mathbf{d}-\mathbf{f}(\mathbf{x}))^T \mathbf{Q}^{-1}(\mathbf{d}-\mathbf{f}(\mathbf{x}))} e^{-\frac{1}{2}(\mathbf{b}-\mathbf{g}(\mathbf{x}))^T \mathbf{R}^{-1}(\mathbf{b}-\mathbf{g}(\mathbf{x}))} \tag{3.10}$$

taking the negative of the log of the likelihood we get the cost function

$$J(\mathbf{x}) = \frac{1}{2}(\mathbf{d} - \mathbf{f}(\mathbf{x}))^T \mathbf{Q}^{-1}(\mathbf{d} - \mathbf{f}(\mathbf{x})) + \frac{1}{2}(\mathbf{b} - \mathbf{g}(\mathbf{x}))^T \mathbf{R}^{-1}(\mathbf{b} - \mathbf{g}(\mathbf{x})) \tag{3.11}$$

where $\mathbf{x} = (\mathbf{m}^T, \mathbf{s}^T)^T$ is the vector of parameters to be estimated, $\mathbf{d}$ and $\mathbf{b}$ are vectors of all odometry and all bearing measurements, and $\mathbf{f}()$ and $\mathbf{g}()$ are all predicted odometry and bearing measurements. The measurement covariances $\mathbf{R}$ and $\mathbf{Q}$ are diagonal since the measurements are independent. The first term in (3.11) penalizes robot motion that does not agree well with odometry measurements and the second term penalizes robot motion and landmark map combinations that do not agree well with bearing measurements.

Taking the first and second derivatives of (3.11) we find

$$\nabla_{\mathbf{x}} J = -\mathbf{J_f}^T \mathbf{Q}^{-1} \epsilon_d - \mathbf{J_h}^T \mathbf{R}^{-1} \epsilon_b \tag{3.12}$$

$$\nabla^2_{\mathbf{x}} J = -\nabla^2_{\mathbf{x}} f \mathbf{Q}^{-1} \epsilon_d + \mathbf{J_f}^T \mathbf{Q}^{-1} \mathbf{J_f} - \nabla^2_{\mathbf{x}} h \mathbf{R}^{-1} \epsilon_b + \mathbf{J_h}^T \mathbf{R}^{-1} \mathbf{J_h} \tag{3.13}$$

where $\mathbf{J_f} = \nabla_{\mathbf{x}} f$ and $\mathbf{J_h} = \nabla_{\mathbf{x}} h$ are the Jacobian matrices of the measurement equations for odometry and bearings, $\epsilon_d = (\mathbf{d} - \mathbf{f}(\mathbf{x}))$ is the difference between measured odometry and odometry predicted by the model parameters, and $\epsilon_b = (\mathbf{b} - \mathbf{g}(\mathbf{x}))$ is the difference between measured and predicted bearings. These last two quantities are the same as the innovations in a Kalman filter.

When the innovations are small the Jacobian inner products dominate expression (3.13); they are also errors which, for an unbiased estimate of $\mathbf{x}$, should tend to cancel out. The Jacobian term will also dominate when the measurement equations are linear (or approximately linear), since $\nabla^2_{\mathbf{x}} f$ or $\nabla^2_{\mathbf{x}} h$ will be zero (or small). In our case, $\mathbf{f}$ is in fact linear, but $\mathbf{h}$ contains an arctangent. Bundle adjustment drops the second derivative terms and approximates the Hessian of the cost function using only the Jacobian inner products, i.e.

$$\mathbf{H} = \nabla^2_{\mathbf{x}} J = \mathbf{J_f}^T \mathbf{Q}^{-1} \mathbf{J_f} + \mathbf{J_h}^T \mathbf{R}^{-1} \mathbf{J_h}. \tag{3.14}$$

Note that $\mathbf{H}$ is the Fisher Information Matrix, or inverse covariance matrix, when $\mathbf{x}$ is the true parameter vector. Newton iterations are used to minimize (3.11) by solving

$$\mathbf{H} \Delta \mathbf{x}_k = -(\mathbf{J_f}^T \mathbf{Q}^{-1} \epsilon_d + \mathbf{J_h}^T \mathbf{R}^{-1} \epsilon_b) \tag{3.15}$$

and then computing the update

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta \mathbf{x}_k \tag{3.16}$$

Generally the complexity of solving an arbitrary system of linear equations in (3.15) would be cubic in the number of model parameters ($3M + 2N$ here). However, photogrammetrists have known for decades how to speed up bundle adjustment using the sparse nature of the structure from motion problem. We can take a similar approach here by exploiting the nature of the bearings-only SLAM problem.

Each bearing measurement depends only on the landmark being measured and the robot pose at the time that the measurement is taken. Therefore, each row of the Jacobian of the bearing measurement equation, $\mathbf{J_h}$, has nonzero entries only for the columns corresponding to the parameters which represent that landmark position and robot pose. The sparse structure is shown in Figure 3.1(a). The Jacobian of the odometry measurement equation, $\mathbf{J_f}$, depends only on two consecutive robot poses. Each row of the Jacobian of the odometry measurement equation, then, has nonzero entries only for the robot poses before and after the motion. Odometry contains no information about landmark positions and corresponding columns of $\mathbf{J_f}$ are zero. The Jacobian has the structure shown in Figure 3.1(b). Because odometry and bearing measurement errors are assumed to be uncorrelated, $\mathbf{Q}$ and $\mathbf{R}$ (and their inverses) are block diagonal. The Hessian $\mathbf{H} = \mathbf{J_f}^T \mathbf{Q}^{-1} \mathbf{J_f} + \mathbf{J_h}^T \mathbf{R}^{-1} \mathbf{J_h}$ has the sparse structure shown in Figure 3.1(c). The upper left is a block tridiagonal matrix U, the lower right is a block diagonal, and the upper right and lower left are rectangular matrices W and W^T.

We can write the equation (3.15) as

$$\begin{bmatrix} U & W \\ W^T & V \end{bmatrix} \begin{bmatrix} \Delta \mathbf{m} \\ \Delta \mathbf{s} \end{bmatrix} = \begin{bmatrix} \epsilon(\mathbf{m}) \\ \epsilon(\mathbf{s}) \end{bmatrix} \tag{3.17}$$

where the Hessian $\mathbf{H}$, the parameter update $\Delta \mathbf{x}$, and the scaled and projected innovations have been partitioned. Now we can premultiply both sides of this

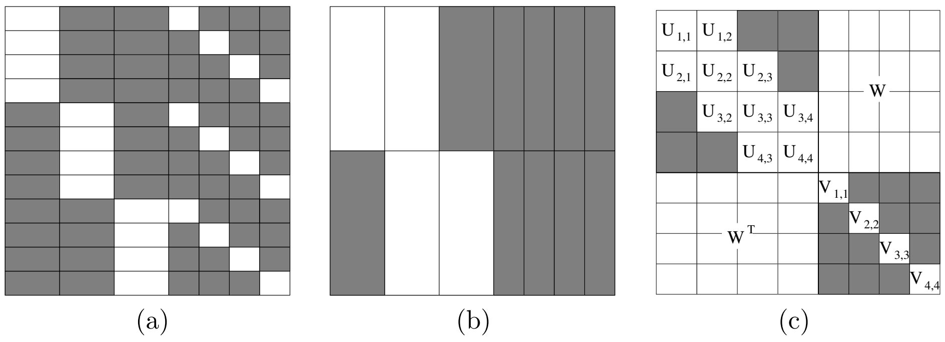


Figure 3.1: Sparse structure of derivatives. (a) Jacobian of bearing measurement equation. (b) Jacobian of odometry measurement equation. (c) Approximation to the Hessian of the cost function.

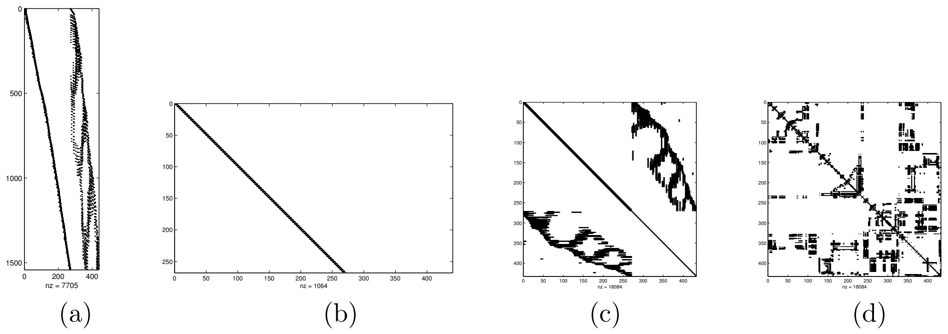


Figure 3.2: Sparsity of (a) the Jacobian of the image measurement function (b) the Jacobian of the odometry measurement function (c) the Hessian for local submap e stimation and (d) the minimum degree reordering for the Hessian.

equation with

$$\begin{bmatrix} I & 0 \\ -W^T U^{-1} & I \end{bmatrix}$$

$$\begin{bmatrix} U & W \\ 0 & V - W^T U^{-1} W \end{bmatrix} \begin{bmatrix} \Delta\mathbf{m} \\ \Delta\mathbf{s} \end{bmatrix} = \begin{bmatrix} \epsilon(\mathbf{m}) \\ \epsilon(\mathbf{s}) - W^T U^{-1} \epsilon(\mathbf{m}) \end{bmatrix} \tag{3.18}$$

We solve (3.18) in two steps. We first solve the bottom equation

$$(V - W^T U^{-1} W)\Delta\mathbf{s} = \epsilon(\mathbf{s}) - W^T U^{-1} \epsilon(\mathbf{m}) \tag{3.19}$$

to find $\Delta\mathbf{s}$ and then substitute it into the top equation, rearranging to get

$$U\Delta\mathbf{m} = \epsilon(\mathbf{m}) - W\Delta\mathbf{s} \tag{3.20}$$

and solve for $\Delta\mathbf{m}$. Computation of $(V - W^T U^{-1} W)$ is $O(MN^2)$, and solving (3.19) is $O(N^3)$. Substituting $\Delta\mathbf{s}$ into (3.20) and solving only requires $O(MN)$

due to the block tridiagonal structure of U, so the overall complexity of bundle adjustment for bearings-only SLAM is $O(MN^2+N^3)$. The complexity of bundle adjustment is larger than that of a Kalman filter, which has a complexity of $O(N^3)$ for N landmarks, but it is not as complex as a general inverse Hessian approach which might require $O((N+M)^3)$.

3.3 Robust estimation

Under the zero mean Gaussian noise assumption, the model parameters which minimize the sum of squared residuals is optimal in the maximum likelihood sense. However, there are many reasons why the noise may not be truly zero mean Gaussian. Spurious features and data association errors can cause introduce a measurement with any arbitrary value within the sensor range, for example $[-\pi, \pi]$ in the case of a bearing measurement. Camera synchronization problems, interlacing artifacts, motion blur, etc. can all cause errors which do not fit the described measurement model.

When errors do not follow the Gaussian noise model assumed, the cost function can be severely affected by these *outliers*. The thin tails of the Gaussian distribution make large noise values extremely rare. These thin tails translate to a disproportionate influence by large discrepancies between observations and expected observations through the square residual terms. Furthermore, large discrepancies result in large contributions to the gradient term, causing the optimization to take large steps in the direction which minimizes the discrepancy due to the outlier.

If we write the normalized residual in the i^{th} measurement as

$$r_i(\mathbf{x}) = \frac{\epsilon_i(\mathbf{x})}{\sigma_i} = \frac{\mathbf{z_i} - h_i(\mathbf{x})}{\sigma_i} \tag{3.21}$$

then least squares seeks to minimize

$$J(\mathbf{x}) = \sum_i (r_i(\mathbf{x}))^2 \tag{3.22}$$

In order to reduce the sensitivity of the estimate to outliers, we may use an M-estimator, in which the squared residual is replaced by a "softer" function

$$J_M(\mathbf{x}) = \sum_i \rho\left(r_i(\mathbf{x})\right) \tag{3.23}$$

where ρ is a symmetric, positive definite function, has a minimum at zero, and is bounded above by r_i^2[42]. The function ρ is chosen so that it increases less dramatically than r_i^2 for large values of r_i. Minimizing (3.23) does not require explicit computation of gradients and Hessians. The problem can instead be translated into an equivalent weighted least squares problem at each step in the iterative optimization. The result is known as *iteratively reweighted least* *squares* (IRLS). The minimum of (3.23) satisfies

$$\sum_i \psi(r_i) \frac{\partial r_i}{\partial \mathbf{x}} = 0 \tag{3.24}$$

where the derivative $\psi(r) = \partial\rho(r)/\partial r$ is called the *influence function*[98]. We can then define a weight function

$$w(r) = \frac{\psi(r)}{r} \tag{3.25}$$

and equation (3.24) becomes

$$\sum_i w(r_i) r_i \frac{\partial r_i}{\partial \mathbf{x}} = 0 \tag{3.26}$$

This is the same system of equations that would be solved in one iteration of the weighted least squares problem

$$\sum_i w(r_i^{(k-1)}) r_i^2 \tag{3.27}$$

where $r_i^{(k-1)}$ indicates the residuals from the last step of the iterative least squares optimization, $r_i(\mathbf{x}^{(k-1)})$.

The influence function $\psi(r_i)$ indicates how much the measurement z_i influences the final estimate. For least squares estimation $\rho(r_i) = r_i^2/2$ and $\psi(r_i) = r_i$, meaning that the influence of the datum grows linearly with the magnitude of the residual, and is unbounded for unbounded error. Table 3.2 shows a list of M-estimators and associated influence functions and weights. This table is adapted from Zhang's tutorial on parameter estimation[98].

The above M-estimators can be used to compute estimates for robot trajectory and feature map parameters from the data. An example is shown below. A synthetic robot trajectory was generated consisting of 80 robot poses in approximately a closed loop. A map containing four landmarks was generated, and measurements were synthesized by computing the expected odometry and bearing measurements and adding Gaussian noise. The optimization was initialized by integrating the odometry to get an initial pose, then triangulating using the trajectory and bearing measurements to get an initial map. The initial trajectory and map were used to initialize an iteratively reweighted least squares algorithm with the six different M-estimators discussed above. The results follow.

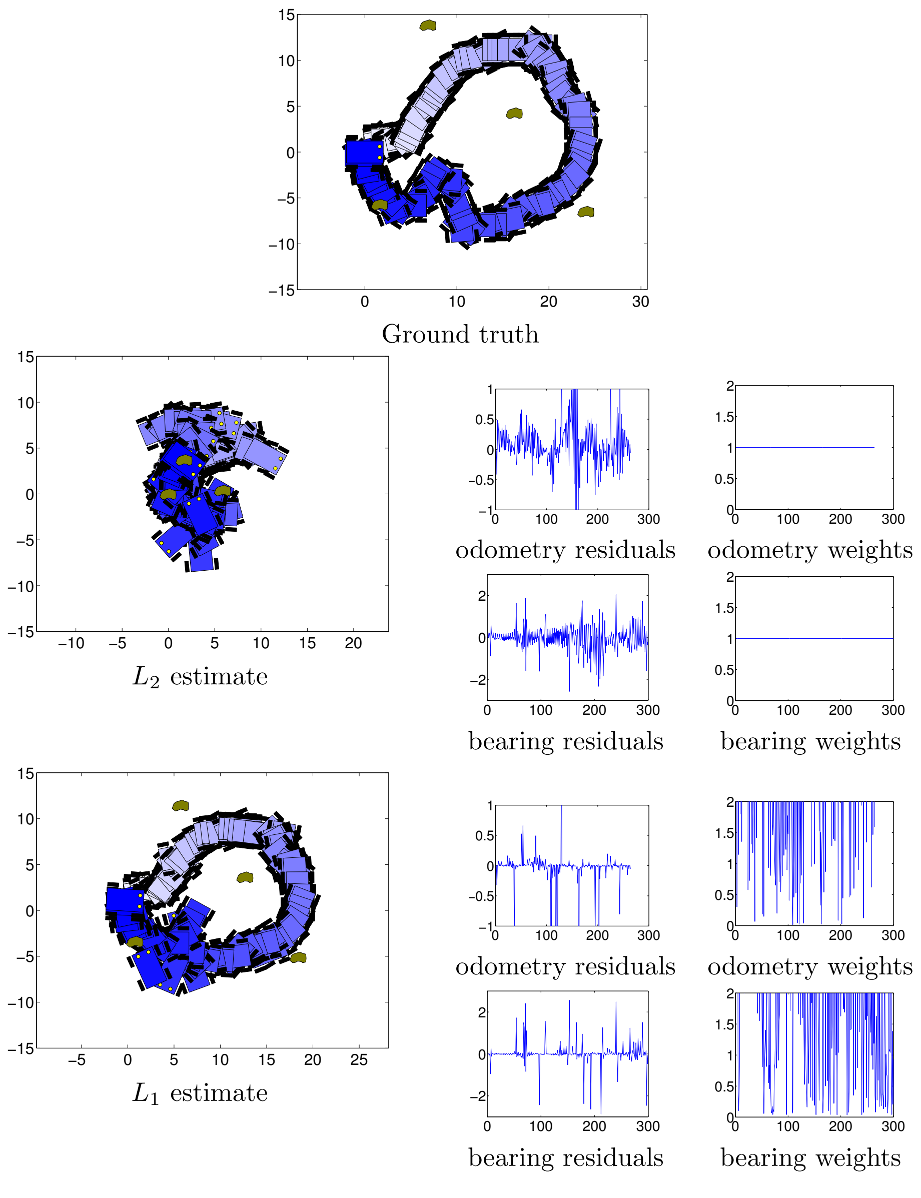


Figure 3.3: L_2 and L_1 robust M-estimators applied to bearings-only SLAM

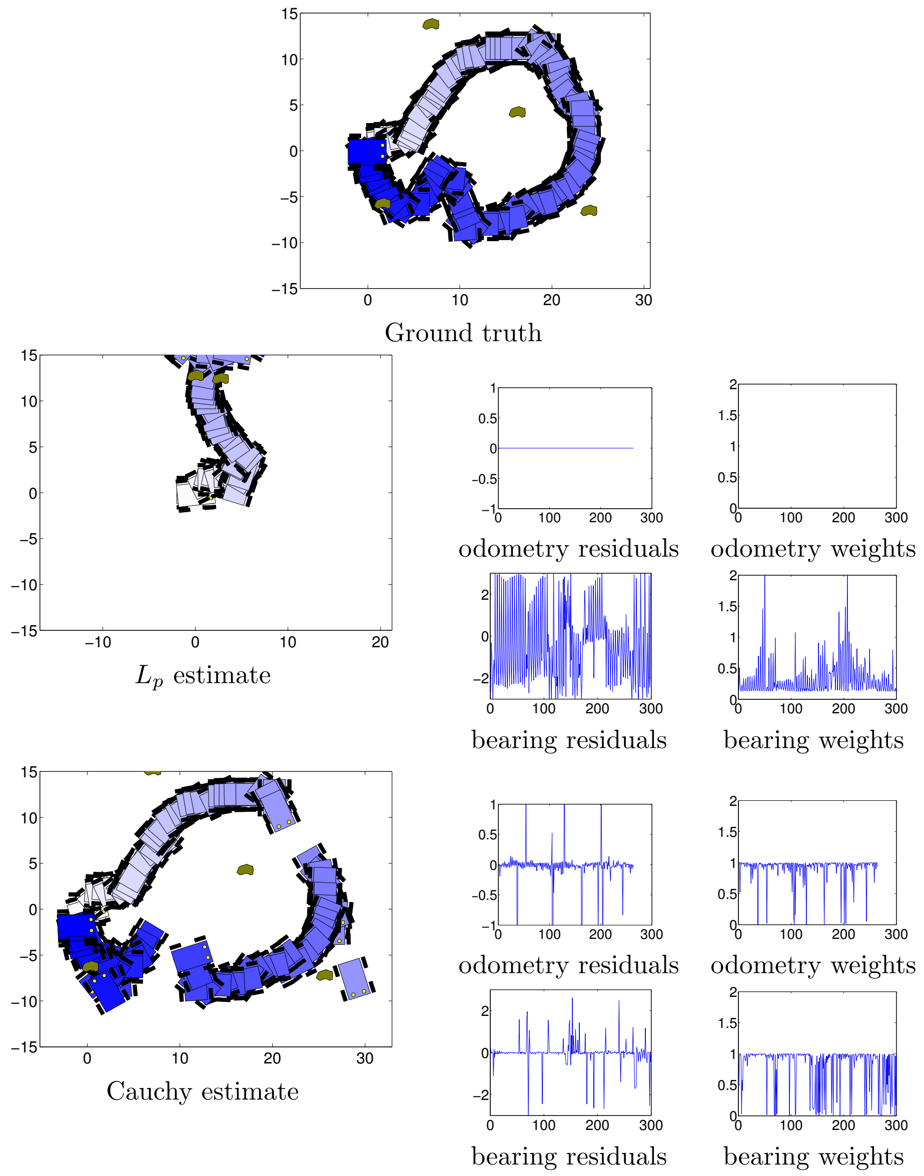


Figure 3.4: L_p and Cauchy robust M-estimators applied to bearings-only SLAM

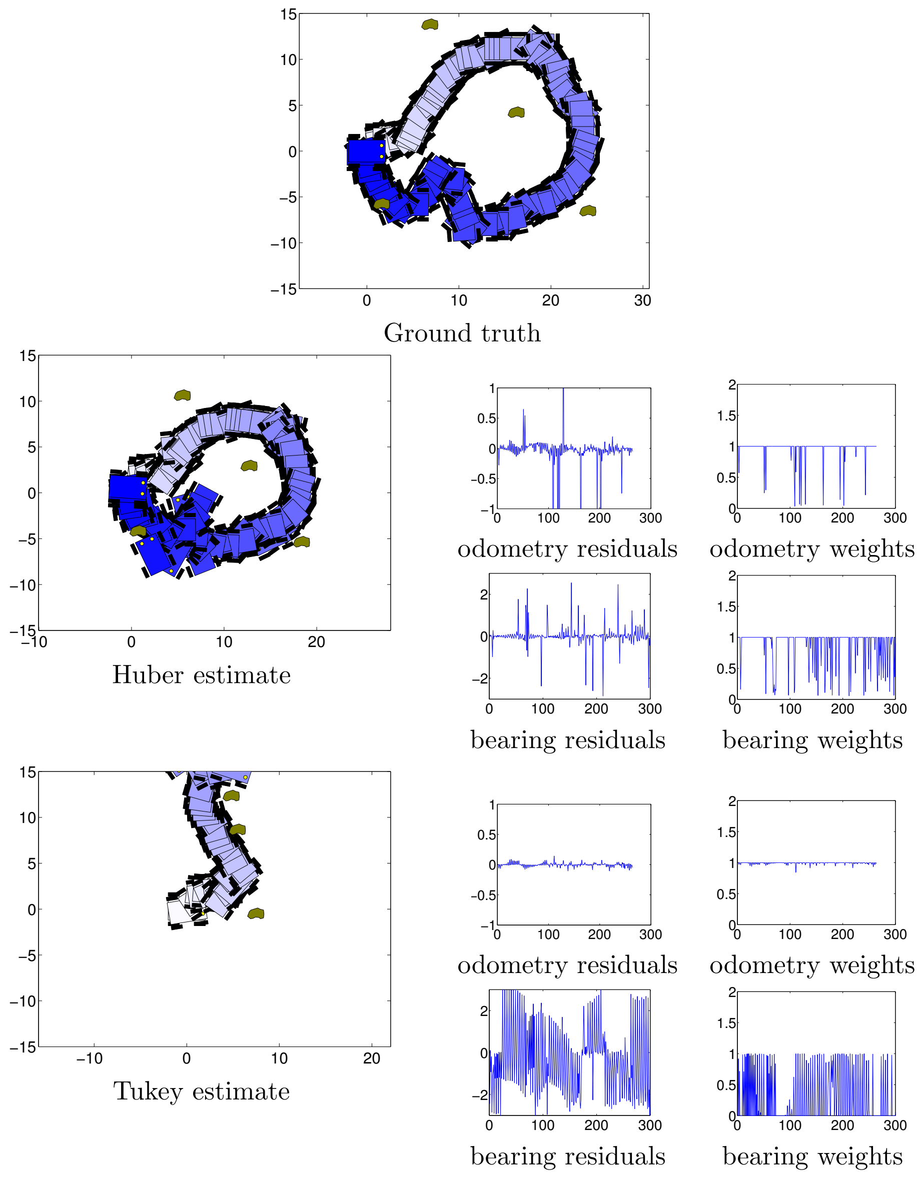


Figure 3.5: Huber and Tukey robust M-estimators applied to bearings-only SLAM

M-estimator	cost $\rho(r)$	influence $\psi(r)$	weight $w(r)$
L_2	$r^2/2$	r	1
L_1	$\lvert r\rvert$	$\mathrm{sgn}(r)$	$\frac{1}{\lvert r\rvert}$
L_p	$\frac{\lvert r\rvert^p}{p}$	$\mathrm{sgn}(r)\lvert r\rvert^{p-1}$	$\lvert r\rvert^{p-2}$
Cauchy	$\frac{c^2}{2}\log(1+(r/c)^2)$	$\frac{r}{1+(r/c)^2}$	$\frac{1}{1+(r/c)^2}$
Huber $\begin{cases}\text{if } \lvert r\rvert \le k \\ \text{if } \lvert r\rvert > k\end{cases}$	$\begin{cases} r^2/2 \\ k(\lvert r\rvert - k/2)\end{cases}$	$\begin{cases} r \\ k\,\mathrm{sgn}(r)\end{cases}$	$\begin{cases} 1 \\ k/\lvert r\rvert\end{cases}$
Tukey $\begin{cases}\text{if } \lvert r\rvert \le c \\ \text{if } \lvert r\rvert > c\end{cases}$	$\begin{cases} (\frac{c^2}{6})(1-[1-(r/c)^2]^3) \\ (c^2/6)\end{cases}$	$\begin{cases} r[1-(\frac{r}{c})^2]^2 \\ 0\end{cases}$	$\begin{cases} [1-(\frac{r}{c})^2]^2 \\ 0\end{cases}$

Table 3.1: Some common M-estimators, adapted from Zhang[98].

3.4 Conclusions

Batch algorithms can be optimal in the sense that by considering all of the model parameters and all of the measurements over the history of the robot, a true maximum a posterior estimate for the trajectory of the rover and the positions of the landmarks. However, this is typically prohibitive for large numbers of robot poses or landmark features.

Projective factorization methods from Structure from Motion are applicable to the bearing only SLAM problem with a slight modification. Unlike the traditional perspective camera model, the measurement model here does not use projection onto a plane to model measurement formation. The typical approach of writing 2D image plane measurements in 3D homogeneous coordinates is modified by writing 1-D bearing measurements z_k as the 2D projective coordinates $(\cos(z_k),\ \sin(z_k))^T$ rather than $(\cot(z_k),\ 1)^T$. The modification has the advantage that rays are never nearly parallel to a virtual image plane, which can cause numerical instability in projective factorization. There is currently no known factorization method for bearing only SLAM that incorporates odometry.

When the measurements are assumed to be corrupted by Gaussian noise, nonlinear least squares optimizes a loss function which is directly related to the Bayes posterior over model parameters given measurements. Nonlinear optimization requires an initial guess for the model parameters. In the bearing only SLAM problem, the odometry measurements can be integrated to generate a guess for the robot trajectory, and a linear triangulation using the initial trajectory and the bearing measurements can provide an initial map. The initial guess is good as long as the odometry is good enough to provide a reliable dead reckoning estimate over the distance traveled by the robot. The method

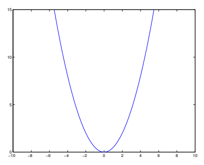
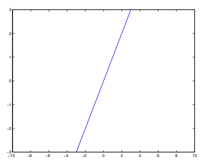
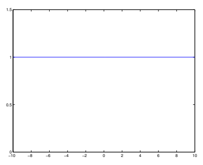
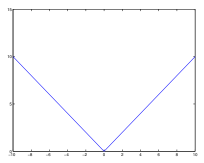
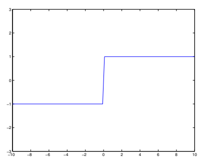
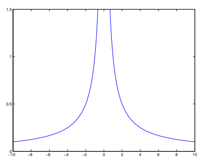
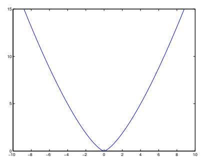
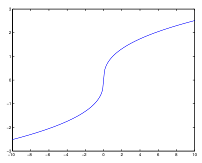
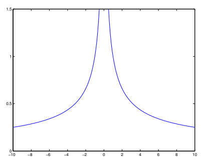
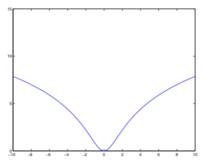
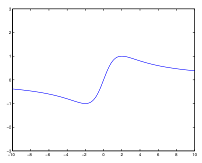
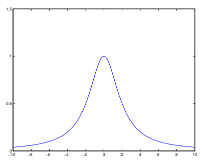
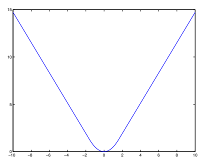
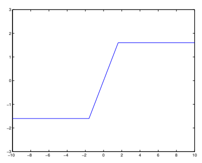
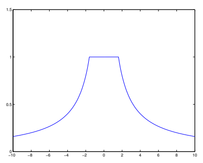
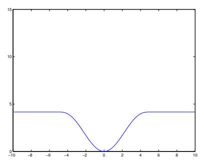
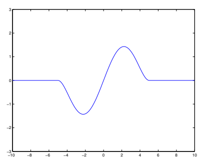
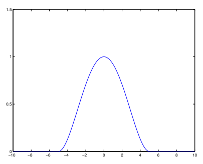
M-estimator	cost $\rho(r)$	influence $\psi(r)$	weight $w(r)$
L_2			
L_1			
L_p			
Cauchy			
Huber			
Tukey			

Table 3.2: M-estimator cost, influence, and weight functions. Adapted from Zhang[98].

converges well for Gaussian noise, but is brittle in the presence of outliers.

When the noise is not Gaussian, or if very unlikely noise corruptions happen, the filter can become unstable. Robust estimators do not suffer from this problem, since the importance of measurements that do not fit the model can be downweighted in order to ignore outliers in the measurements. Several robust estimators were presented here and applied to an example bearing-only SLAM problem. The L_2 estimator is the same as nonlinear least squares and does not perform well on data with outliers. The L_p estimator also failed to perform well on data including outliers. The primary reason seems to be that the weight function itself diverges for small residuals, which can cause a few measurements which fit well to dominate the cost function. This is particularly problematic if one measurement is used to provide an initial guess for the parameters, since that measurement will have zero residual (to machine precision). The L_1 and Huber estimators perform well on the example problem shown.

The nature of these batch algorithms provide important insight into how recursive estimators work. A good recursive estimator is designed to closely approximate the batch estimator. The next chapter will discuss how Kalman filters and related algorithms accomplish this approximation.

Chapter 4

Hierarchical estimation

In many Structure from Motion problems, the camera motion and scene structure are recovered in one large estimation or optimization step. Figure 4.1(a) shows a diagram of this strategy. A hierarchical approach attempts to reconstruct the scene using a divide-and-conquer scheme. The first step is to break the image sequence into smaller subsequences and recover motion and structure within that subsequence. At the higher level, the models generated from subsequences are registered and merged together to form an estimate of the structure of the entire scene. This approach is shown in Figure 4.1(b). The goal of the hierarchical approach is to reduce the total computation required for map building without compromising too much on the resulting map quality. The degree to which resulting map quality is compromised depends on how much information is retained in the submap estimation and representation.

Hierarchical estimation can refer to two slightly different methods for breaking the SLAM problem into smaller problems, solving each one independently, and combining the solutions from each subproblem to produce a single, global solution. The hierarchy may occur in space or in time, although typically the two are tightly coupled.

Spatial hierarchy refers to mapping algorithms which produce submaps by any means. Batch methods[10] or recursive methods[34, 16] can be used for producing the submaps. Maintaining consistency within the spatial hierarchy involves knowing when the robot leaves a submap and enters a different submap, and whether this submap is a new region or one which has already been visited. This by itself can be a difficult problem.

Temporal hierarchy refers to breaking the observation stream into shorter substreams and producing an independent map and trajectory estimate for each substream. These maps may not be spatially disjoint, which means that consistency requires the registration algorithm to find overlapping regions and to find unique position estimates for features which occur in multiple submaps.

Hierarchical mapping approaches also differ in whether they enforce global geometric constraints after the local maps are constructed. Chong and Kleeman's hierarchical mapping algorithm only preserves topological information

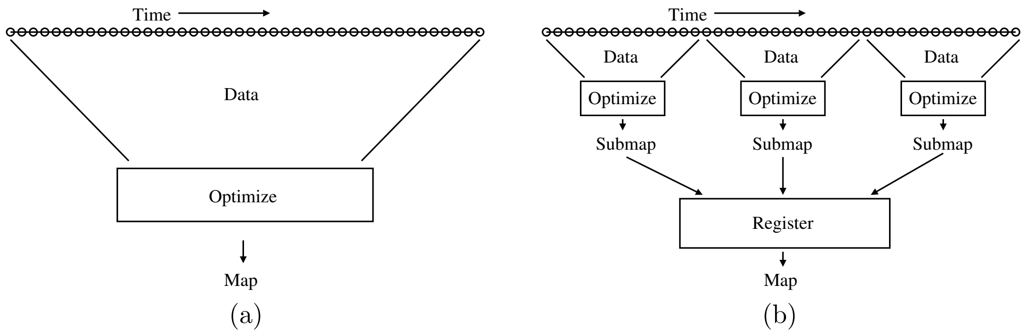


Figure 4.1: (a) Full optimization approach to structure and motion recovery (b) Hierarchical approach to structure and motion recovery

about the connection between submaps[16]. Burgard et al. use the EM algorithm to globally register submaps produced with sonar while trusting odometry to produce locally accurate trajectory estimates[10]. Lu and Milios[50] align range scans directly, using odometry to provide an initial guess for the alignment and treating each range scan as a local submap. The Decoupled Stochastic Map (DSM) algorithm of Feder and Leonard[49] aligns submaps using the vehicle motion from one region to another in order to register submaps relative to neighboring submaps during transitions[34].

Relative filtering algorithms are closely related to hierarchical methods. In relative filtering, local relationships are estimated which are invariant to coordinate frame. The algorithm can produce a map consisting only of local relationships[22], or use global constraints to produce a globally consistent map using the local invariant relationships[65, 25]. These methods are an extreme form of the divide and conquer philosophy, where each submap is simply a relationship between the smallest number of features which still contain some information. If the heading of the vehicle is known, then the difference in x and y coordinates can be estimated for any pair of points[22] and a set of linear constraints can be applied to enforce global consistency[65]. If the heading is not known, then the invariant relationships consist of pairwise distances between two landmarks or angles subtended by two features at a third feature, and global consistency requires solving a nonlinear optimization problem[25].

A hierarchical SFM approach is presented in [36] which builds submodels using trilinear constraints, then merges submodels and recovers scene and motion parameters. However, in a model building or mapping application, the recovery of the camera parameters for every view may not be necessary and a considerable amount of computation could be eliminated. Another hierarchical SFM approach is presented in [78] which computes local structure estimates and then projects them into virtual “key frames”, eliminating the viewing parameters from the submodel estimation but also projecting the submodel into a small number of virtual views. It is not clear whether the key frames capture as much information about the reconstructed model as possible given the original

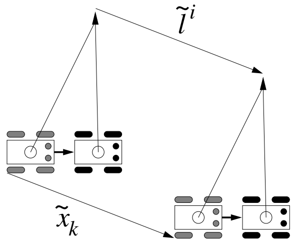


Figure 4.2: Errors in estimates of rover pose and landmark position, in the global frame, are correlated

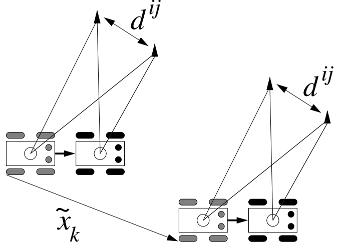


Figure 4.3: Environment features invariant to rover pose are impervious to errors in pose estimate

data.

Hierarchical estimation has characteristics in common with recursive and batch approaches. By breaking up the environment map into smaller local submaps, the algorithm needs to consider only some of the data at a time. Limiting the amount of data that needs to be processed at once can reduce the processing requirements.

4.1 Invariant filtering

In any realistic scenario, the pose of the robot is not known to absolute certainty. The estimate $\hat{x}_k$ of the robot pose has some associated error $\tilde{x}_k = \hat{x}_k - x_k$ and landmark position estimates $\hat{\ell}^i$ have error $\tilde{\ell}^i = \hat{\ell}^i - \ell^i$. One source of complexity for absolute coordinate frames in SLAM is that landmark position estimates $\hat{\ell}^i$ and the robot pose estimate $\hat{x}_k$ are correlated. Assuming for the moment that the bearing measurements z_k^i and z_{k+1}^i are perfect, the error in the landmark position due to the error in robot pose can be found by expanding $\tilde{\ell}^i$ then taking a Taylor expansion

$$\begin{aligned}\tilde{\ell}^i &= \hat{\ell}^i - \ell^i \\ &\quad \mathcal{L}(\mathbf{x}_k + \tilde{\mathbf{x}}_k, \mathbf{u}_k, \mathbf{z}_k^i, \mathbf{z}_{k+1}^i) - \\ &\quad \mathcal{L}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{z}_k^i, \mathbf{z}_{k+1}^i)\end{aligned}$$

$$\tilde{\ell}^i \approx (\nabla_{\mathbf{x}_k}\mathcal{L})\tilde{\mathbf{x}}_k + O((\tilde{\mathbf{x}}_k)^2) \tag{4.1}$$

Dropping all but the first order term,

$$E\{\tilde{\mathbf{x}}_k\tilde{\ell^i}_k\} = E\{\tilde{\mathbf{x}}_k(\nabla_{\mathbf{x}_k}\mathcal{L})\tilde{\mathbf{x}}_k\} \tag{4.2}$$

Figure 4.2 shows a diagram of how this happens. The vehicle measures the bearing to a target, moves, and measures the bearing to the target again. The position of the target is measured in a rover-centered coordinate frame with the initial robot position as its origin. The resulting estimate of the landmark location in absolute coordinates depends on where the vehicle was when it began its motion and the error in landmark position is directly correlated with the error in robot position.

A similar argument demonstrates how the error in the robot pose estimate becomes correlated to errors in the map when the rover attempts to localize relative to landmarks with erroneous locations. Over time, filtering methods such as Kalman filters which recursively update robot pose and landmark positions in an absolute coordinate frame can diverge.

In addition, absolute filtering approaches to solving the SLAM problem suffer the curse of dimensionality. The state space is of dimension N where in a planar environment, N is three parameters for the robot pose plus two parameters for every landmark. Kalman filters require $O(N^2)$ storage and $O(N^3)$ computation per step in order to keep track of the full covariance matrix, and as the robot explores larger and larger regions, the filter becomes intractable.

The fundamental importance of the relative filter[22] is that it recursively updates estimates of environmental features which are invariant to the robot pose. Consider the distance between two landmarks,

$$d^{ij} = ||\ell^i - \ell^j||^2. \tag{4.3}$$

The estimate $\hat{d}^{ij}$ and error $\tilde{d}^{ij}$ do not depend on $\mathbf{x}_k$. If $\mathcal{D}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{z}^i_k, \mathbf{z}^j_k, \mathbf{z}^i_{k+1}, \mathbf{z}^j_{k+1})$ is the estimator for d^{ij}, then

$$\nabla_{\mathbf{x}_k}\mathcal{D}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{z}^i_k, \mathbf{z}^j_k, \mathbf{z}^i_{k+1}, \mathbf{z}^j_{k+1}) = 0 \tag{4.4}$$

and therefore

$$E\{\tilde{\mathbf{x}}_k\tilde{d}^{ij}_k\} = 0. \tag{4.5}$$

Even an error in heading, which is notoriously problematic for dead reckoning, does not affect the estimate. Figure 4.3 shows a robot estimating the distance d^{ij} between landmarks i and j. The position of each landmark is wrong due to $\tilde{x}_k$ but the relationship between the landmarks is unaffected.

The two invariant features which can be estimated are pairwise distances, as mentioned above, and angles ψ^{ijk} subtended by a landmark pair ℓ^i, ℓ^k as measured from a third, ℓ^j, as in Figure 4.4. The endpoints which define a distance, or the triplet of points which define an angle, are called *control points.* For simplicity, we will deal with only distance measurements for the rest of this paper, but note that angles may be more difficult to use. Angles represent more nonlinear relationships and may present singularities. In addition, estimating angles requires the simultaneous observation of three landmarks instead of two.

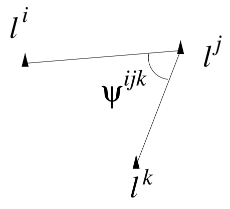


Figure 4.4: Angles between two landmarks subtended at a third are invariant to robot pose

4.1.1 Data structure

The invariant filter requires two separate “maps” to be maintained. The first map is a relative map similar to that of [22]. This map contains the invariant relationships between landmarks. Each invariant feature has an associated type (distance or angle), a list of the landmarks which make up its control points, and a mean and covariance for the estimated feature value.

The second map is an absolute map which contains estimates for landmark positions in a global coordinate frame. Each landmark in the absolute map contains an estimated (x, y) coordinate and a list of the invariant features for which it is a control point. This enables efficient computation of the derivatives for optimizing the absolute map with respect to the relative map. Figure 4.5 shows a diagram of these data structures and their relationships.

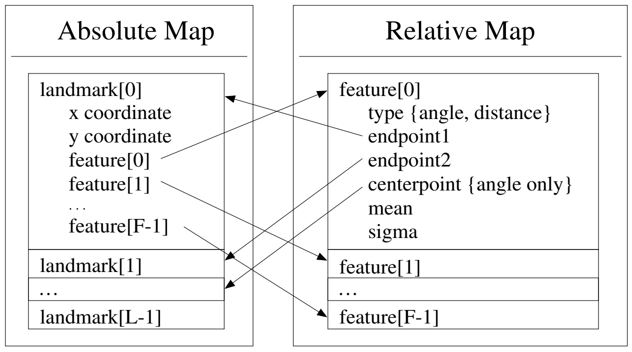


Figure 4.5: Relationship between absolute map and relative map data structures

4.1.2 Building the maps

When a new landmark is encountered, its position is estimated in the global reference frame using the estimate of the robot pose and an estimate of the landmark position in the robot coordinate frame. The landmark is then inserted into the absolute map at that location. It is only important that the landmark be inserted close enough to its "true" position that the optimization procedure used to update the absolute map converges to the right solution.

Invariant features involving the new landmark must also be inserted into the relative map. A greedy triangulation algorithm is used to connect the new landmark to observable neighbors. Note that we do not require a planar triangulation. The triangulation must simply provide enough constraints to fix the landmarks in the absolute map. An attempt is made to create invariant features using landmarks which are close together, since landmarks which are far apart are less likely to be observed simultaneously.

4.1.3 Strategy for avoiding correlations

The goal in this filtering method is to ensure that the state vector is decoupled so that memory requirements and update computation are trivial. In order to do this, we must be careful not to introduce correlations. Correlation between robot pose error and landmark position errors is eliminated by using map features which are invariant to robot pose. By proper consideration, we can also eliminate correlation between map features.

When bearing observations are made, the bearings $Z_k = \{z_k^i; i = 0, \ldots\}$ to all visible targets at time k are stored. The measurement u_k of motion from time k to $k+1$ is also stored.

Triangulation allows computation of the landmark positions using the motion estimate as a baseline,

$$\hat{\ell}^i = \mathcal{L}(\mathbf{x}_k, \mathcal{F}(\mathbf{x}_k, \mathbf{u}_k), \mathbf{z}_k^i, \mathbf{z}_{k+1}^i). \tag{4.6}$$

We then compute the Euclidean distance between pairs of landmarks as $\hat{d}^{ij} = ||\hat{\ell}^i - \hat{\ell}^j||^2$. The error $\tilde{d}^{ij}$ will be correlated with $\mathbf{u}_k$, $\mathbf{z}_k^i$, $\mathbf{z}_{k+1}^i$, $\mathbf{z}_k^j$, and $\mathbf{z}_{k+1}^j$.

In order to be sure that the estimate $\hat{d}^{ij}$ is not correlated with any other features in the relative map, we cannot use any of $\{\mathbf{u}_k, \mathbf{z}_k^i, \mathbf{z}_{k+1}^i, \mathbf{z}_k^j, \mathbf{z}_{k+1}^j\}$ to estimate other features. The restriction that we cannot use $\mathbf{z}_{k+1}^i$ and $\mathbf{z}_{k+1}^j$ means that we must remove those measurements from the buffer. The restriction that we cannot use $\mathbf{u}_k$ means that not only will $\mathbf{u}_k$ be removed from the buffer, but also all old bearing measurements $\mathbf{z}_k$ as well, since these bearings cannot be used for triangulation without estimates of motion after the bearings were measured. The result is that after step $k+1$, only $\mathbf{z}_{k+1} - \{\mathbf{z}_{k+1}^i, \mathbf{z}_{k+1}^j\}$ remain in the measurement buffer for the next step.

4.1.4 Updating the relative map

At time k, the algorithm checks the buffer to find all pairs of consecutive bearing measurements of the same landmark $\{\mathbf{z}_{k-1}^i, \mathbf{z}_k^j\}$. These pairs create a list of observable invariant features which are checked against the invariant features already in the relative map. This check is necessary because not every landmark pair distance or landmark triplet angle appear in the relative map.

Once a list of candidate features is found, a decision must be made as to which feature should be updated. We have adopted a greedy strategy where the observable relative feature $\hat{d}_k^{ij}$ with the highest covariance σ_k^{ij} is updated. The selected feature is updated using the familiar equations

$$\hat{d}_{k+1}^{ij} = \frac{(\sigma_{\mathcal{D}})^2 \hat{d}_k^{ij} + (\sigma_k^{ij})^2 \mathcal{D}(\hat{\ell}_{k+1}^i, \hat{\ell}_{k+1}^j)}{(\sigma_{\mathcal{D}})^2 + (\sigma_k^{ij})^2} \tag{4.7}$$

$$\sigma_{k+1}^{ij} = \frac{(\sigma_{\mathcal{D}})^2 (\sigma_k^{ij})^2}{(\sigma_{\mathcal{D}})^2 + (\sigma_k^{ij})^2} \tag{4.8}$$

where $\hat{d}_k^{ij}$ and σ_k^{ij} are the estimate and variance at time k and $\mathcal{D}(\hat{\ell}_{k+1}^i, \hat{\ell}_{k+1}^j)$ is the new measurement, with variance $\sigma_{\mathcal{D}}$.

4.1.5 Updating the absolute map

The update of the absolute map is independent of the update of the relative map and can be done asynchronously. The upate step can be invoked automatically whenever the relative map is updated or can be delayed until a localization is required, a relative feature estimate changes dramatically, or some other criterion.

The update is done by optimizing the relationships between landmarks in the absolute map with respect to the relationships estimated in the relative map. This is done by applying Levenberg-Marquardt (LM) optimization to the cost function

$$J = \sum_{p=0}^{F_k} \left(\frac{\hat{d}^{i(p)j(p)} - \mathcal{D}(\hat{\ell}^{i(p)}, \hat{\ell}^{j(p)})}{\sigma^{i(p)j(p)}} \right)^2 \tag{4.9}$$

where F_k is the number of features in the relative map at time k and $i(p)$ and $j(p)$ are the indices for the landmarks which are the control points for the p^{th} feature.

The LM algorithm has an advantage over the EKF or IEKF (i.e. Newton-Raphson) update in cases where inverse Hessian methods cause iterations to diverge away from the solution. The LM algorithm checks the solution at each step by evaluating (4.9) and if the solution is worse, LM smoothly transitions to gradient descent behavior by forcing the Hessian to be diagonally dominant[71]. One step of LM has the same complexity as one step of the EKF or IEKF.

4.1.6 Memory and computational requirements

At time k, let L_k be the number of landmarks in the absolute map, F_k be the number of features in the relative map and V_k be the number of landmarks that are visible. In the worst case, i.e. for small areas or long range sensors where every landmark is visible at each step, V_k could be L_k, but generally $V_k < L_k$. With a reasonable triangulation, F_k is $O(L_k)$.

Checking for new features to add is fast. A list is maintained containing all of the landmarks which require another invariant feature in order to be properly

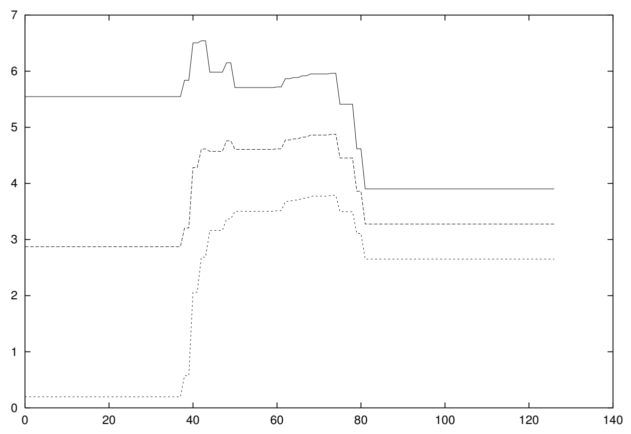


Figure 4.6: Convergence of relative feature 0, d^{01}.

constrained. This list is usually much smaller than L_k. Determining whether a new landmark is to be added is $O(V_k)$, since all visible landmarks must be checked to see if any are new. Determining which invariant feature to update is $O(F_k)$ in the worst case. Once the feature to be added or updated is selected, the update is $O(1)$. $O(F_k)$ memory is required to store the relative map since a full covariance matrix is not needed.

The update of the absolute map given the features in the relative map is done using Levenberg-Marquardt optimization. One iteration is $O(L_k^3)$ in general, but may be faster if sparse matrix techniques are used to speed up the inversion of the Hessian. The algorithm has quadratic convergence near the optimum, so although we require an unknown number of iterations there should not be many if the solution is near the optimum when iteration begins. We also do not need to update the absolute map every cycle. $O(L)$ memory is required to store the absolute map.

By comparison, Kalman filtering requires an $O(L_k^3)$ update each time new observations become available, and $O(L_k^2)$ memory to maintain the covariance matrix.

4.1.7 Example

The above algorithm has been implemented and tested in simulation and on real data. In the interest of time we will only present results with real data here.

The robot is an RWI ATRV with an omnidirectional camera onboard. The robot logs images from the camera at about 1Hz, and odometry at about 10 Hz. The "landmarks" are large textured foam blocks. It is not our intent at this time to solve the tracking or landmark recognition problem, so the landmark selection in the omnidirectional images was done by hand, and the visual tracking requires a user to identify targets.

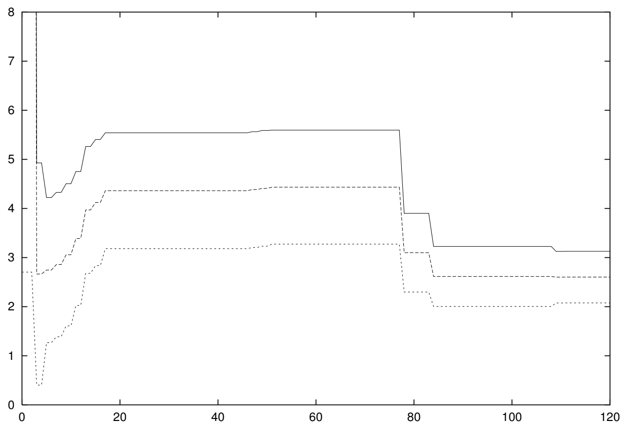


Figure 4.7: Convergence of invariant feature 4, d^{23}.

During these tests, the ATRV had a broken wheel. The result was a significant bias in dead reckoning which was partially calibrated out. The error model was also modified to increase the vehicle motion uncertainty to a much higher degree when going through large turns. The invariant filtering algorithm successfully recovers the relationships between the landmarks even with this bias.

Figure 4.6 and 4.7 show the time evolution of invariant features d^{01} and d^{23}, the first and fourth features to be added to the relative map. The estimated mean is shown along with $3\sigma^{ij}$ intervals.

Ground truth for the map is not available. Instead, we implemented a full minimum mean square error (MMSE) batch estimation and used all available data to optimize the estimate of map and robot trajectory. Since the map computed by the invariant filter is not computed with an absolute position reference, there is an unknown planar transformation between the map from the invariant filter and the map from the MMSE estimate. This is accounted for by finding the transformation which most closely aligns the two maps.

The map recovered by the invariant filter is shown in Figure 4.8 along with the MMSE map. The landmark positions agree to within about a half meter after registration.

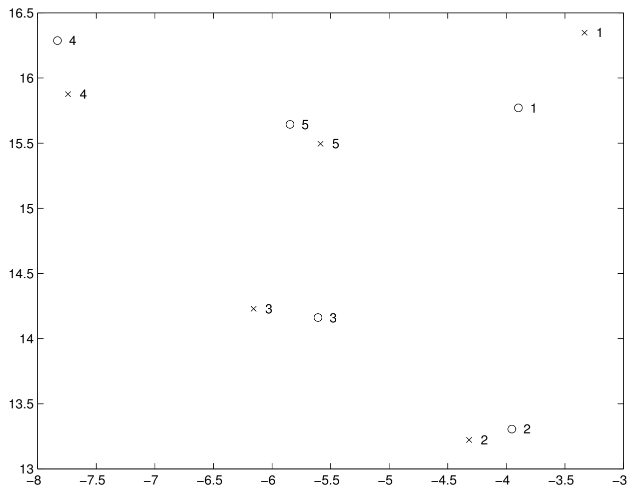


Figure 4.8: Final estimated map (x's) with MMSE estimate (o's)

4.2 Temporal hierarchical mapping

In this section we will show a hierarchical mapping scheme which breaks the observation stream into temporal subsequences. Each subsequence is processed in batch and produces a local submap. The submaps may or may not be spatially distinct. After the submaps are computed, they are all registered to a common coordinate frame, again using batch estimation.

4.2.1 Submap estimation

We would like to be able to produce a maximum likelihood estimate for the robot motion and scene structure given a subset of the bearing and odometry measurements, denoted $\mathbf{z}_k$ and $\mathbf{d}_k$. In order to do this, we must find the robot trajectory $\hat{\mathbf{m}}_k$ and environment map $\hat{\mathbf{x}}_k$ which maximize the likelihood $p(\mathbf{b}_k, \mathbf{d}_k|\mathbf{m}_k, \mathbf{x}_k)$. We will drop the batch index subscript k for clarity. From the noise model described above, we have

$$p(\mathbf{z}, \mathbf{d}|\mathbf{m}, \mathbf{x}) = p(\mathbf{z}|\mathbf{m}, \mathbf{x})p(\mathbf{d}|\mathbf{m}) = \prod_{i,j} p(z_{ij}|\mathbf{m}_i, \mathbf{x}_j) \prod_i p(\mathbf{d}_i|\mathbf{m}_{i-1}, \mathbf{m}_i) \quad (4.10)$$

That is, each image measurement z_{ij} depends on the position of landmark j and the pose of the robot at time i, while each odometry measurement $\mathbf{d}_i$ depends on the pose of the robot at time $i-1$ and time i, before and after the motion measured. Taking the negative log of the likelihood yields (up to scale)

$$\mathcal{L}(\mathbf{b}, \mathbf{d}|\mathbf{m}, \mathbf{x}) = (\mathbf{b} - \mathbf{g}(\mathbf{m}, \mathbf{x}))^T \mathbf{R}^{-1} (\mathbf{b} - \mathbf{g}(\mathbf{m}, \mathbf{x})) + (\mathbf{d} - \mathbf{f}(\mathbf{m}))^T \mathbf{Q}^{-1} (\mathbf{d} - \mathbf{f}(\mathbf{m})) \quad (4.11)$$

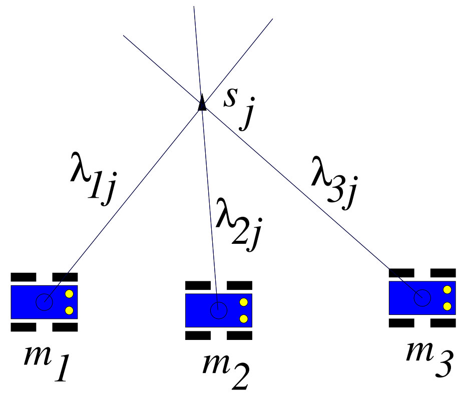


Figure 4.9: Given robot positions and bearing measurements, the landmark position can be computed as a linear least squares problem

which must be minimized to find $\hat{\mathbf{m}}, \hat{\mathbf{x}}$. This cost function is similar to that of bundle adjustment in SFM and the minimization can be done efficiently using Levenberg-Marquardt (LM)[90, 71, 24]. Taking the Jacobian $\mathcal{J} = \nabla\mathcal{L}$ and using the Gauss-Newton approximation [71] to the Hessian $\mathcal{H} \approx \nabla^2\mathcal{L}$ of the cost function above, we compute

$$\begin{aligned} \mathcal{J} &= -\mathbf{H}^T\mathbf{R}_z^{-1}(\mathbf{z} - \mathbf{h}(\mathbf{m}, \mathbf{x})) - \mathbf{F}^T\mathbf{R}_d^{-1}(\mathbf{d} - \mathbf{f}(\mathbf{m})) \qquad (4.12) \\ \mathcal{H} &= \mathbf{H}^T\mathbf{R}_z^{-1}\mathbf{H} + \mathbf{F}^T\mathbf{R}_d^{-1}\mathbf{F} \qquad (4.13) \end{aligned}$$

where H and F are the Jacobian of $h()$ and $f()$ respectively. The Gauss-Newton update to the parameters is computed using equations (4.13)

$$\begin{aligned} \left(\Delta\mathbf{m}^T, \Delta\mathbf{x}^T\right)^T &= (\mathcal{H})^{-1}\mathcal{J} \qquad (4.14) \\ \mathbf{m}^{i+1} &= \mathbf{m}^i + \Delta\mathbf{m} \qquad (4.15) \\ \mathbf{x}^{i+1} &= \mathbf{x}^i + \Delta\mathbf{x} \qquad (4.16) \end{aligned}$$

This process iterates until convergence to a locally optimum submap and sub-trajectory estimate.

4.2.2 Initial submap

As a nonlinear optimization method, LM requires a good initial parameter estimate in order to converge to a good solution. Dead reckoning, although inaccurate over long distances, can be used to provide an initial estimate of the trajectory of the robot over short distances. Once an initial trajectory estimate is available, it can be held fixed and the landmark position estimation can be solved as a linear least squares problem. The geometry is shown in Figure 4.9. We introduce an unknown variable λ_{ij} which is the distance from the robot at time i to landmark j. The geometry is such that

$$\begin{aligned} s_{xj} &= m_{xi} + \lambda_{ij}\cos(m_{\theta i} + z_{ij}) \\ s_{yj} &= m_{yi} + \lambda_{ij}\sin(m_{\theta i} + z_{ij}) \end{aligned} \tag{4.17}$$

These constraints are rearranged to be of the form

$$\begin{pmatrix} 1 & 0 & -\cos(\phi_{1j}) & 0 & \cdots & 0 \\ 0 & 1 & -\sin(\phi_{1j}) & 0 & \cdots & 0 \\ 1 & 0 & 0 & -\cos(\phi_{2j}) & \cdots & 0 \\ 0 & 1 & 0 & -\sin(\phi_{2j}) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & 0 & 0 & 0 & \cdots & -\cos(\phi_{Mj}) \\ 0 & 1 & 0 & 0 & \cdots & -\sin(\phi_{Mj}) \end{pmatrix} \begin{pmatrix} s_{xj} \\ s_{yj} \\ \lambda_{1j} \\ \lambda_{2j} \\ \vdots \\ \lambda_{Mj} \end{pmatrix} = \begin{pmatrix} m_{x1} \\ m_{y1} \\ m_{x2} \\ m_{y2} \\ \vdots \\ m_{xM} \\ m_{yM} \end{pmatrix} \tag{4.18}$$

where $\phi_{ij} = m_{\theta i} + z_{ij}$. Every bearing measurement of landmark j provides two linear equations and one additional unknown, the depth λ_{ij}. Each pair of equations from one observation z_{ij} is linearly independent, and the equations generated from robot poses that are not colinear with the landmark are also linearly independent. Solving (4.18) finds the least squares estimate for the landmark position and all of the depth parameters.

Once a reasonable initial guess is available for both the robot trajectory and the environment map, the nonlinear optimization algorithm can refine the estimate for both.

4.2.3 Map registration and merging

At the higher level of the hierarchy, the submaps that are found using the nonlinear optimization described above must be registered. We treat the map that is recovered as a measurement W of some section of the full map from an uncertain location. That is

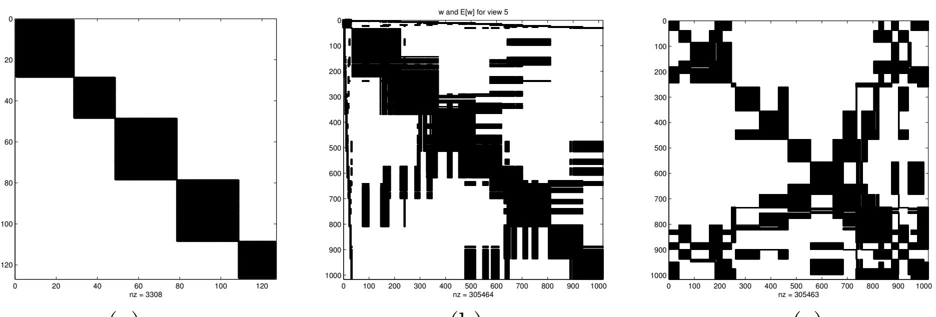
$$\mathbf{s}_i = \mathbf{P}_i\mathbf{s} + \xi_i \tag{4.19}$$

where $\mathbf{s}$ is the true map, $\mathbf{P}_i$ is a permutation matrix which selects the landmarks from the full map that are present in submap i, and ξ_i is noise with covariance $\mathbf{Q}_i$.

Each submap is related to the global map by some unknown similarity transformation $\mathbf{T}_i$. The submaps $\mathbf{s}_i$ we would expect to measure given a similarity transformation $\hat{\mathbf{T}}_i$, a map $\hat{\mathbf{s}}$, and a permutation matrix $\mathbf{P}_i$ is simply $E\{\mathbf{s}\} = \hat{\mathbf{T}}_i\mathbf{P}_i\hat{\mathbf{s}}$, so we define the squared error cost function

$$J(\hat{\mathbf{T}}, \hat{\mathbf{s}}) = \sum_i (\mathbf{s}_i - \hat{\mathbf{T}}_i\mathbf{P}_i\hat{\mathbf{s}})^T\mathbf{Q}_i^{-1}(\mathbf{s}_i - \hat{\mathbf{T}}_i\mathbf{P}_i\hat{\mathbf{s}}) \tag{4.20}$$

with hats denoting quantities to be estimated while $\mathbf{s}_i$ (the submap) and $\mathbf{P}_i$ (the permutation, or correspondence) are known. The similarity transform $\mathbf{T}_i$



(a) (b) (c)

Figure 4.10: (a) Measurement covariance for registration and merging step. (b) Hessian for registration/merging. (c) Reordered Hessian.

is of the form

$$\mathbf{T}_i = \alpha_i \begin{pmatrix} cos(\theta_i) & -sin(\theta_i) & \Delta x_i \\ sin(\theta_i) & cos(\theta_i) & \Delta y_i \end{pmatrix} = \begin{pmatrix} a_i & -b_i & c_i \\ b_i & a_i & d_i \end{pmatrix} \tag{4.21}$$

By substituting the unknown parameters $\{a_i, b_i, c_i, d_i\}$ for the unknown parameters $\{\alpha_i, \Delta x_i, \Delta y_i, \theta_i\}$, our cost function is bilinear in the unknown parameters $\mathbf{T}_i$ and $\mathbf{s}$. Minimizing equation (4.20) will find an optimal global map $\mathbf{s}^*$ given the submaps and their covariances computed at the lower level.

Once again, there is sparseness which can be exploited in solving (4.20). The measurement covariance is block diagonal, since each submap was computed independently. The measurement covariance matrix is shown in Figure 4.10(a). The Jacobian of the measurement function for the registration stage is only nonzero in row i and column j if feature j was mapped in submap i. Less overlap between submaps will mean more sparseness in the Hessian, although more overlap between submaps will result in stronger connections between adjacent submaps and a better reconstruction. The LM Hessian approximation for (4.20) has the structure shown in Figure 4.10(b). The LM Hessian can be reordered using a minimum degree ordering in order to speed up the computation of the LM update step, an example of a reordered Hessian is shown in Figure 4.10(c).

4.2.4 Example

An example problem is shown in Figure 4.11 where a robot moves in a pattern which covers a large rectangular area in short, roughly parallel rows. The dead reckoning accumulates significant errors over the course of the motion, but locally each section of the trajectory estimate from dead reckoning is accurate enough to initialize search for a local map and trajectory estimate. Figure 4.12 shows the submaps estimated for sections of the trajectory, and Figure 4.13 shows the resulting global map after the local submaps are registered together.

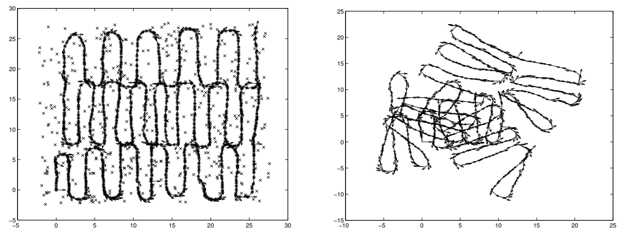

(a) (b)

Figure 4.11: (a) Ground truth trajectory and map (b) Dead reckoning trajectory estim ate

4.3 Conclusions

Invariant filtering methods decouple the robot position and orientation from the map in order to simplify computation. The strength of the invariant filtering approach to SLAM lies in the decoupling of the estimates of each feature relationship so that the map states are completely uncorrelated with each other and the robot state. This allows a much more scalable approach since the computational and memory requirements scale linearly with the number of features. A more recent formulation of the relative mapping philosophy can be found in[85].

As a solution to SLAM, hierarchical mapping is more of a meta-approach than a full solution to the problem. There still needs to be some way of computing local submaps, and the method chosen for submaps needs to satisfy constraints of algorithmic complexity, memory requirements, and robustness to outliers. The hierarchical mapping approach might use batch or recursive methods to build submaps, and once the submaps are built the system can either use geometric constraints to register the maps or leave them loosely but topologically connected. The registration algorithm described in this thesis is an efficient method for aligning point sets under similarity transforms which is linear in the parameters. By posing the registration problem so that it is linear in the parameters, we can avoid local optima and guarantee an optimal solution, subject to numerical considerations.

In some ways hierarchical mapping strikes a balance between full map techniques such as the original stochastic mapping algorithm[80], and relative or invariant filtering techniques[22, 65, 25]. Relative filtering techniques choose to estimate the smallest meaningful geometric relationships between features, such as pairwise distances between features, and then build a network of such geometric constraints to map the environment. Hierarchical mapping instead uses a metric map of the local region as its primitive, and then places that region submap in a greater context in the global map using loose topological information or by registering the submaps. Global mapping uses the entire world map as its primitive, rather than breaking the map into smaller primitives.

The question of which representation to use has implications on the computational complexity, memory requirements, and speed of an algorithm, and the quality and usefulness of the resulting map. However, there is a deeper question behind which representation and which approach is the best to use. The real question is which representation does the best job of capturing the information that is present in the data. On way to answer this question is to look at the quality of the maps produced by each algorithm and rank them based on residual mapping error. Another way is to look at the question from first principles, and find an approximation which most closely matches the true posterior. Answering the representation question in this way proves to be a difficult question, but can lead to insights into how to best capture the information in the data in a recursive algorithm.

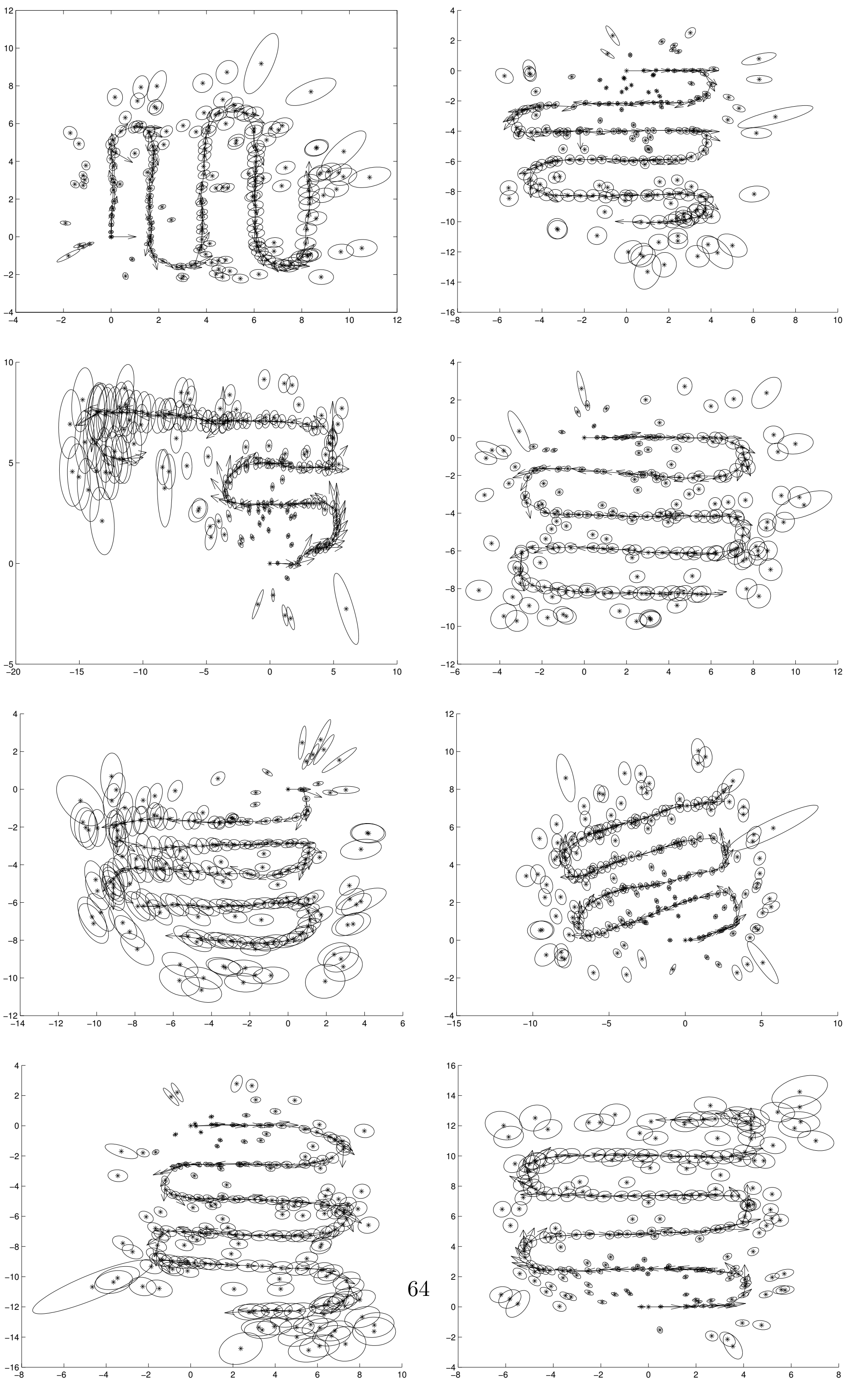


Figure 4.12: Recovered submaps

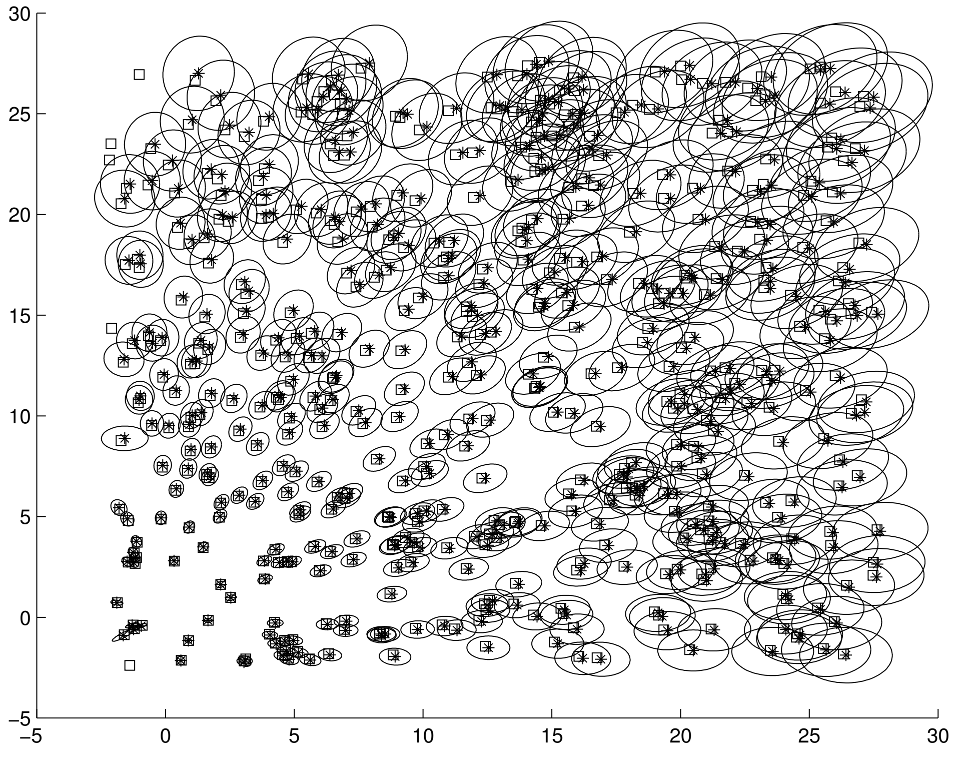


Figure 4.13: Reconstructed full map (*'s) with 99% uncertainty ellipses and ground truth (squares)

Part III

Recursive estimation

Chapter 5

Recursive estimation

Recursive estimation is a natural approach for Markov systems. In general, recursive approaches are an attempt to process observations as they become available in order to determine a posterior state estimate, or a posterior density over state space. The observations are then discarded and the filter retains only the posterior distribution itself. This approach is necessary because of the on-line constraints faced by real world systems, and the enormous number of measurements that can be made over an operational lifetime.

If the distribution used by the filter can represent everything that can be determined about the state given all of the observations made so far, then recursive estimation can provide the same performance as full maximum likelihood estimation at a lower cost. A familiar example where this is possible is Kalman filters applied to linear systems with Gaussian noise. However, when the distribution cannot represent everything that can be determined from the data, an approximation is necessary.

Recursive estimation is a data reduction technique. The data reduction occurs in two ways. The first is computation of statistics from observations. The statistics are then used in place of the data in order to make inferences. Inferences can be made more efficiently using the statistics than using the original data. The second reduction is in the marginalization of state variables. In a Kalman filter for dynamic systems, an assumption is made that the past history of the dynamic state is not necessary given an estimte of the current state. In SLAM, the robot pose is estimated over time, but if the entire history of vehicle motion is not of interest then all but the most recent pose can be discarded.

In linear systems, the mean and covariance computed by the Kalman filter are a sufficient statistic. For this reason the Kalman filter is an optimal filter for linear Gaussian processes. For nonlinear problems the statistics are not sufficient, and several approaches for computing statistics which best capture the information in the data are described in this chapter. Chapter 6 presents a new method which is based on some of the ideas in this chapter.

5.1 General Markov systems

In a general Markov system, the belief state can be updated from time step $t-1$ to time step t in the following way. A system model is required which describes the distribution for the current state given the previous state, or $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. A measurement model is also required, which describes the probability of making a particular measurement given a system state, or $p(\mathbf{z}_t|\mathbf{x}_t)$. If we begin with a probability over state space at time $t-1$ which is conditioned on all of the measurements up to that time, then the state at time t can be expressed as[1]

$$p(\mathbf{x}_t|\mathbf{z}_t) = \int p(\mathbf{x}_t|\mathbf{z}_t, \mathbf{x}_{t-1})p(\mathbf{x}_{t-1})d\mathbf{x}_{t-1} \tag{5.1}$$

We can then apply Bayes' rule to the first probability in the integral to get

$$p(\mathbf{x}_t|\mathbf{z}_t) \propto \int p(\mathbf{z}_t|\mathbf{x}_t, \mathbf{x}_{t-1})p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1})d\mathbf{x}_{t-1} \tag{5.2}$$

The current measurement $\mathbf{z}_t$ is assumed to be independent of the previous state or previous measurements given the current state so that $p(\mathbf{z}_t|\mathbf{x}_t, \mathbf{x}_{t-1}) = p(\mathbf{z}_t|\mathbf{x}_t)$. We now have

$$p(\mathbf{x}_t|\mathbf{z}^t) \propto \int p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1})d\mathbf{x}_{t-1} \tag{5.3}$$

The last factor in the integral is the belief state at time $t-1$, and this gives us a recursive algorithm for updating the belief at each time step.

5.2 Kalman filter

The Kalman filter represents the belief state using only the first two central moments of the distribution, the mean and covariance. Given the mean and covariance at one time, the Kalman filter proceeds by making predictions of the state at the next time step using the current state, then updating the estimate using new observations. The mathematics behind the predict and update equations follow directly from the Gaussian assumption and the process and measurement models.

The Kalman filter maintains an estimate of the state which is parameterized in terms of a mean and covariance. These form a sufficient statistic for the posterior if it is Gaussian. The state at time t is a random variable $\mathbf{x}_t \sim p(\mathbf{x}_t)$, and the estimated mean and covariance are

$$\begin{aligned} \hat{\mathbf{x}}_t &= E_p\{\mathbf{x}_t\} \\ \widehat{\mathbf{C}}_t &= E_p\{(\mathbf{x}_t - \hat{\mathbf{x}}_t)(\mathbf{x}_t - \hat{\mathbf{x}}_t)^T\} \end{aligned} \tag{5.4}$$

[1]The prior $p(\mathbf{x}_{t-1}|\mathbf{z}^{t-1})$ is actually conditioned on past measurements. Since the old measurements are not used for inference, we drop the $\mathbf{z}^{t-1}$ for convenience and write $p(\mathbf{x}_{t-1})$.

The notation $\hat{\mathbf{x}}_{t|t-1}, \widehat{\mathbf{C}}_{t|t-1}$ denotes the prediction, or the estimate of the state at time t using measurements up to time $t-1$. The notation $\hat{\mathbf{x}}_{t|t}, \widehat{\mathbf{C}}_{t|t}$ denotes the update, or the estimate of the state at time t using the measurement from time t.

5.2.1 Prediction

The dynamic model is used to predict the state of the system at the next time step. We can generally write the dynamic model as

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t; \mathbf{u}_t) + \omega_t \tag{5.5}$$

where $\mathbf{u}_t$ is some control input and ω_t is the process noise. The control input is a constant, which we will drop in the sequel for simplicity. The process noise captures all of the unpredictable or unmodeled effects and is assumed to be zero mean Gaussian noise with covariance Q_t. The zero mean assumption is not restrictive since if $E_p\{\omega_t\}$ were nonzero we could account for that in the process model. Using the predictive model and the assumption that the state at time t is Gaussian distributed, we can write the expected state, or prior mean

$$\begin{aligned}\hat{\mathbf{x}}_{t+1} &= E_p\{\mathbf{x}_{t+1}\} \\ &= E_p\{\mathbf{f}(\mathbf{x}_t) + \omega_t\} \\ &= E_p\{\mathbf{f}(\mathbf{x}_t)\} + E_p\{\omega_t\}\end{aligned} \tag{5.6}$$

Since the process noise is zero mean,

$$\hat{\mathbf{x}}_{t+1} = \int p(\mathbf{x}_t) f(\mathbf{x}_t) d\mathbf{x}_t \tag{5.7}$$

The prior covariance is

$$\begin{aligned}\widehat{\mathbf{C}}_{t+1} &= E_p\{(\mathbf{x}_{t+1} - \hat{\mathbf{x}}_{t+1})(\mathbf{x}_{t+1} - \hat{\mathbf{x}}_{t+1})^T\} \\ &= E_p\{(\mathbf{f}(\mathbf{x}_t) + \omega_t - \hat{\mathbf{x}}_{t+1})(\mathbf{f}(\mathbf{x}_t) + \omega_t - \hat{\mathbf{x}}_{t+1})^T\} \\ &= E_p\{(\mathbf{f}(\mathbf{x}_t) - \hat{\mathbf{x}}_{t+1})(\mathbf{f}(\mathbf{x}_t) - \hat{\mathbf{x}}_{t+1})^T\} \\ &\quad +2E_p\{(\mathbf{f}(\mathbf{x}_t) - \hat{\mathbf{x}}_{t+1})\omega_t^T\} + E_p\{\omega_t \omega_t^T\}\end{aligned} \tag{5.8}$$

The second term disappears since the process noise is uncorrelated with the state prediction. The third term is the process noise covariance Q_t, so the prior covariance is

$$\widehat{\mathbf{C}}_{t+1} = \int p(\mathbf{x}_t)(\mathbf{f}(\mathbf{x}_t) - \hat{\mathbf{x}}_{t+1})(\mathbf{f}(\mathbf{x}_t) - \hat{\mathbf{x}}_{t+1})^T d\mathbf{x} + \mathbf{Q}_t \tag{5.9}$$

With linear predictive models, the state equation is a linear function. In the EKF and IEKF, the prediction model is linearized about the current state estimate using a first-order Taylor expansion, i.e. computing the Jacobian of the process model and then treating the model as though it were linear. If the state prediction model can be written as

$$\mathbf{f}(\mathbf{x}_t) \approx \mathbf{f}(\hat{\mathbf{x}}_t) + \mathbf{F}(\mathbf{x}_t - \hat{\mathbf{x}}_t) \tag{5.10}$$

where the matrix $\mathbf{F} = \frac{\partial}{\partial \mathbf{x}}\mathbf{f}$ then the predicted mean in (5.6) becomes

$$\begin{aligned}
\hat{\mathbf{x}}_{t+1} &= E_p\{\mathbf{f}(\hat{\mathbf{x}}_t) + \mathbf{F}(\mathbf{x}_t - \hat{\mathbf{x}}_t) + \omega_t\} \\
&= E_p\{\mathbf{f}(\hat{\mathbf{x}}_t)\} + \mathbf{F}E_p\{\mathbf{x}_t - \hat{\mathbf{x}}_t\} + E_p\{\omega_t\} \\
&= \mathbf{f}(\hat{\mathbf{x}}_t) \qquad (5.11)
\end{aligned}$$

and the predicted variance in (5.8) becomes

$$\begin{aligned}
\widehat{\mathbf{C}}_{t+1|t} &= E_p\{(\mathbf{f}(\hat{\mathbf{x}}_{t|t}) + \mathbf{F}(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t}) - \hat{\mathbf{x}}_{t+1|t})(\mathbf{f}(\hat{\mathbf{x}}_{t|t}) + \mathbf{F}(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t}) - \hat{\mathbf{x}}_{t+1|t})^T\} \\
&\quad + 2E_p\{(\mathbf{f}(\hat{\mathbf{x}}_{t|t}) + \mathbf{F}(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t}) - \hat{\mathbf{x}}_{t+1|t})\omega_t^T\} + E_p\{\omega_t\omega_t^T\} \\
&= E_p\{\mathbf{F}(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t})(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t})^T\mathbf{F}^T\} + \mathbf{Q}_t \\
&= \mathbf{F}\widehat{\mathbf{C}}_{t|t}\mathbf{F}^T + \mathbf{Q}_t \qquad (5.12)
\end{aligned}$$

5.2.2 Update

We are given a prior mean and covariance for the state variable x_t from the prediction step. Assuming that the state is Gaussian distributed, these statistics fully describe the Normal distribution

$$p(\mathbf{x}_t) = \mathcal{N}(\hat{\mathbf{x}}_t, \widehat{\mathbf{C}}_t). \tag{5.13}$$

This distribution is conditioned on measurements $\{z_0 \dots z_{t-1}\}$ but we drop the conditional distribution notation for simplicity. After acquiring a new measurement z_t, we would like to update our estimate $\hat{\mathbf{x}}_t$. The measurement is given by some function of the current state (the measurement equation) plus observation noise

$$\mathbf{z}_t = \mathbf{h}(\mathbf{x}_t) + \nu_t \tag{5.14}$$

The observation noise ν_t is zero mean with covariance $\mathbf{R}_t$. The zero mean assumption is not restrictive since if $E_p\{\nu_t\}$ is nonzero it can be accounted for in $\mathbf{h}$. From the observation model,

$$p(\mathbf{z}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{h}(\mathbf{x}_t), \mathbf{R}_t). \tag{5.15}$$

The update step of the Kalman filter is a maximum a posterior (MAP) estimation problem. The goal of MAP estimation is to find $\hat{\mathbf{x}}_t^*$ which maximizes $p(\mathbf{x}_t|\mathbf{z}_t)$, or

$$\hat{\mathbf{x}}_t^* = \text{argmax}_{\mathbf{x}_t}\, p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t) \tag{5.16}$$

Maximizing (5.16) is equivalent to minimizing an objective function given by the negative log of the posterior. For Gaussian distributions this is a quadratic form

$$\begin{aligned}
J(\mathbf{x}_t) &= -Log(p(\mathbf{z}_t|\mathbf{x}_t)) - Log(p(\mathbf{x}_t)) \\
&= (\mathbf{z}_t - \mathbf{h}(\mathbf{x}_t))^T\mathbf{R}_t^{-1}(\mathbf{z}_t - \mathbf{h}(\mathbf{x}_t) + (\mathbf{x}_t - \hat{\mathbf{x}}_t)\widehat{\mathbf{C}}_{t|t-1}^{-1}(\mathbf{x}_t - \hat{\mathbf{x}}_t) \qquad (5.17)
\end{aligned}$$

We can find the minimum of $J(\mathbf{x}_t)$ using Gauss-Newton. Gauss-Newton requires the first and second derivatives of J with respect to $\mathbf{x}$,

$$\begin{aligned}\nabla_{\mathbf{x}} J(\mathbf{x}) &= -\mathbf{H}_t^T \mathbf{R}^{-1}(\mathbf{z} - \mathbf{h}(\mathbf{x})) + \widehat{\mathbf{C}}_{t|t-1}^{-1}(\mathbf{x} - \hat{\mathbf{x}}) & (5.18)\\ \nabla_{\mathbf{x}}^2 J(\mathbf{x}) &= \mathbf{H}_t^T \mathbf{R}^{-1}\mathbf{H}_t + \widehat{\mathbf{C}}_{t|t-1}^{-1} & (5.19)\end{aligned}$$

where $\mathbf{H}_t$ is the Jacobian of the measurement function $\mathbf{h}(\mathbf{x})$ and we have made the assumption that $\nabla_{\mathbf{x}}^2 \mathbf{h}(\mathbf{x})$ is small.

The Hessian of J is $-\nabla^2 log(p(\mathbf{x}_t|\mathbf{z}_t))$ or the Fisher information matrix. The Fisher information is the inverse of the covariance of the updated estimate, so

$$\widehat{\mathbf{C}}_{t|t}^{-1} = \mathbf{H}_t^T \mathbf{R}^{-1}\mathbf{H}_t + \widehat{\mathbf{C}}_{t|t-1}^{-1} \tag{5.20}$$

which is the well known covariance update equation for the Kalman filter.

Gauss-Newton minimization of (5.17) requires iteratively updating $\mathbf{x}$ by an amount δ given by

$$\nabla_{\mathbf{x}}^2 J \; \delta = -\nabla_{\mathbf{x}} J \tag{5.21}$$

or

$$\delta = -(\nabla_{\mathbf{x}}^2 J)^{-1} \nabla_x J. \tag{5.22}$$

Using (5.17), we find

$$\delta = \widehat{\mathbf{C}}_{t|t} \left(\mathbf{H}_t^T \mathbf{R}_t^{-1}(\mathbf{z}_t - \mathbf{h}(\mathbf{x}_t)) - \widehat{\mathbf{C}}_{t|t-1}^{-1}(\mathbf{x}_t - \hat{\mathbf{x}}_t) \right) \tag{5.23}$$

The updated state estimate $\hat{\mathbf{x}}_{t|t}$ then becomes

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \widehat{\mathbf{C}}_{t|t} \left(\mathbf{H}_t^T \mathbf{R}_t^{-1}(\mathbf{z}_t - \mathbf{h}(\mathbf{x}_t)) - \widehat{\mathbf{C}}_{t|t-1}^{-1}(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t-1}) \right). \tag{5.24}$$

The terms in (5.24) can be interpreted as follows. The first term pulls the state estimate toward a value of the state which would produce the measurement that was observed, or a zero measurement residual state. The second term pulls the state estimate toward the prior mean estimate. The degree to which the solution is pulled toward the prior is $\mathbf{C}_{t|t-1}$, so a small prior covariance will pull strongly toward the prior mean and a large prior covariance will not. The degree to which the solution is pulled toward the zero measurement residual is $\mathbf{H}_t^T \mathbf{R}_t^{-1}$, so a measurement with very small variance will pull strongly toward the zero residual state, and a measurement with large covariance will not. The derivative of the measurement equation is also important. If the measurement function is locally flat, the measurement will not strongly influence the estimated mean and if the measurement function is locally sensitive to the state, the measurement can have a strong influence.

The iterated extended Kalman filter is implemented by repeating (5.20) and (5.24) until they converge, to find an locally optimal estimate for the mean and covariance. The extended Kalman filter simply computes each quantity once, implicitly assuming that one iteration of Gauss-Newton will find an estimate that is close enough to the local optimum. If the measurement equation is linear, then Gauss-Newton is guaranteed to converge in one step, and the Kalman filter is optimal. When the measurement function is nonlinear, we cannot expect convergence in one step. The IEKF repeats the Gauss-Newton iterations until convergence. Convergence can be determined by computing the step size for one Gauss-Newton iteration and terminating when the step is small relative to the state covariance. The number of iterations depends on the size of the innovation, the complexity of the nonlinearities, and the threshold for the convergence criterion. The threshold can be interpreted as a χ^2 statistic, so that its value has some statistical meaning.

5.2.3 Kalman filtering and bearing only SLAM

The EKF has been used extensively in SLAM where range measurements are available. For the bearing only problem, the EKF is less well suited. The biggest problem lies in initializing new features. Because one measurement is not sufficient for initializing a new feature, several problems arise. The first problem is initializing the location of the landmark. With a range-bearing sensor, the landmark location can be initialized in the map coordinates by transforming the landmark position in robot coordinates to global coordinates using the current estimated robot pose. With only one bearing measurement, the landmark location is still unknown. The approach used in the experiments in this thesis is to initialize the landmark at some arbitrary distance away from the rover in the direction along the bearing measurement. The insertion of a landmark into the state vector is then

$$\mathbf{x} \leftarrow \begin{pmatrix} \mathbf{x} \\ m_x + \alpha \cos(z + m_\theta) \\ m_y + \alpha \sin(z + m_\theta) \end{pmatrix} \tag{5.25}$$

where m_x, m_y, m_θ are the pose of the robot. The insertion of a landmark into the map also requires the addition of rows and columns of the covariance matrix. Since there is no prior knowledge of the landmark location, the covariance is unbounded. Using the information form of the filter, zero entries in the information matrix can properly capture this (lack of) knowledge. However, in the covariance form, we must instead inserting a large covariance for the new landmark in order to keep the covariance matrix nonsingular. In a Bayesian sense, inserting the landmark with a finite covariance is equivalent to putting a prior on the location of the landmark which is given by the normal distribution $\mathcal{N}(\mathbf{x}_j, \mathbf{P}_{jj})$. With a large enough covariance, the hope is that the first few observations will "swamp the prior," and that this prior is simply for numerical stability. Inserting a new landmark, then, requires a change to the covariance matrix of

$$\mathbf{C} \leftarrow \begin{pmatrix} \mathbf{C} & 0 \\ 0 & \mathbf{P}_{jj} \end{pmatrix} \tag{5.26}$$

Once the landmark is inserted into the state vector and covariance matrix, the filter procedes with updating the state using the measurement.

Empirically, the EKF does poorly for bearing-only SLAM with any considerable level of noise in the observations. The IEKF can be used to overcome the problems of the EKF when there is a moderate amount of noise. Primarily, the EKF is insufficient because posterior statistics are computed for observations before the state has been disambiguated, and these posterior statistics are not a good approximation to the original measurement when the state estimation errors are large. The EKF also has trouble minimizing the nonlinear cost function in one step. The repeated Gauss-Newton iterations of the IEKF can overcome this latter issue.

5.3 Derivative free state estimation

There has been a lot of interest recently in state estimation techniques which do not require the analytic computation of Jacobians[44, 46, 18, 68, 94, 95]. There are many advantages to eliminating the Jacobian computation from a recursive filter. Dynamic and measurement models may not be analytic. If they are analytic they may not be differentiable, e.g. functions may be discontinuous. If they are analytic and differentiable, getting all of the terms correct can be a hassle if the models are large or complicated. Furthermore, there is interest in the automatic software generation community of providing tools for the creation of state estimators[9] Providing a tool which requires only a user-supplied state transition function and user-supplied measurement function is advantageous. The user can more quickly and easily get the filter up and running.

For these reasons, one might consider a derivative free approach in order to provide a trade-off between simplicity of implementation and filter performance. However, in reality, it is possible to outperform Jacobian based filters with derivative free filters. It's a win-win situation; one can avoid computing and implementing Jacobians *and* do better than Kalman filters at the same time.

In this section we will discuss the theory behind derivative free state estimation and review two methods from the literature.

There are two major differences in approaches to derivative free state estimation. The first has to do with whether the samples are used to interpolate the model equations and produce effective model linearizations, or to compute posterior statistics for the squared error term directly. The second difference has to do with how 1-dimensional sampling rules are extended to multiple dimensions. We will begin with a brief overview of two existing methods.

5.3.1 Divided difference filter

The Divided difference filter (DDF) [68] replaces the Jacobian computation in the EKF with a central divided difference. Rather than computing the analytic Jacobian of the function $\mathbf{h}(\mathbf{x})$, the filter computes a polynomial interpolation of $h(x)$ using Stirling's formula.

$$h'(x) = h_0 + \frac{\mu\delta h_0}{\Delta x}(x - x_0) + \frac{1}{2}\frac{\delta^2 h_0}{(\Delta x)^2}(x - x_0)^2 \tag{5.27}$$

where

$$\begin{aligned}\mu\delta h_0 &= \frac{1}{2}(h(x_0+\Delta x) - h(x_0-\Delta x)) \\ \delta^2 h_0 &= h(x_0+\Delta x) - 2h(x_0) + h(x_0-\Delta x) \qquad (5.28)\end{aligned}$$

Consideration is given to the appropriate choice of Δx, which is found to be $\sqrt{3}\sigma$ where σ^2 is the variance of the posterior.

The divided difference filter computes the Jacobian $\mathbf{H}$ of the interpolated polynomial in (5.27), which is a best first order approximation to the nonlinear model $\mathbf{h}()$. The linear interpolation is used to compute an approximation to ∇e and $\nabla^2 e$ by

$$\begin{aligned}\nabla e &\approx -\mathbf{H}^T\mathbf{R}^{-1}(\mathbf{z} - \mathbf{h}(\mathbf{x})) \\ \nabla^2 e &\approx \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} \qquad (5.29)\end{aligned}$$

However, if $\mathbf{h}()$ contains significant second order terms, this linearization is not sufficient. To compensate for this, the authors have developed a second order divided difference algorithm. Compared with the Unscented filter which will be described below, the performance of the second order DDF is about the same but the DDF algorithm is more complex. However, the DDF algorithm and its development are interesting because the method approaches linearization in recursive state estimation from a numerical interpolation and differentiation perspective, rather than from a numerical integration point of view.

5.3.2 Unscented filter

The Unscented filter [92] uses a deterministic sampling scheme to compute the posterior mean and covariance in (5.7) and (5.9) numerically. The integral is computed using a set of weighted samples, replacing the analytic integral with a numerical sum.

The samples used in the sum are chosen to match the moments of the Gaussian distribution. Consider the standard normal in one dimension. We wish to approximate the Gaussian with three samples located on the real line at $\mathcal{X} = \{x_0, x_1, x_2\}$ and with weights $\mathcal{W} = \{w_0, w_1, w_2\}$. With three weighted samples, we have six degrees of freedom. The degrees of freedom correspond to the location of the three samples on the real line, and the weight associated with each one. These six degrees of freedom can be used to match the six constraints which require the empirical moments of the sample set to match the first five moments of the standard normal, and for the weights to sum to one,

$$\begin{aligned} & \textstyle\sum_i w_i &= 1 \\ \mu = & \textstyle\sum_i w_i x_i &= 0 \\ \sigma^2 = & \textstyle\sum_i w_i (x_i - \mu)^2 &= 1 \\ s = & \textstyle\sum_i w_i (x_i - \mu)^3 &= 0 \\ k = & \textstyle\sum_i w_i (x_i - \mu)^4 &= 3 \end{aligned}$$

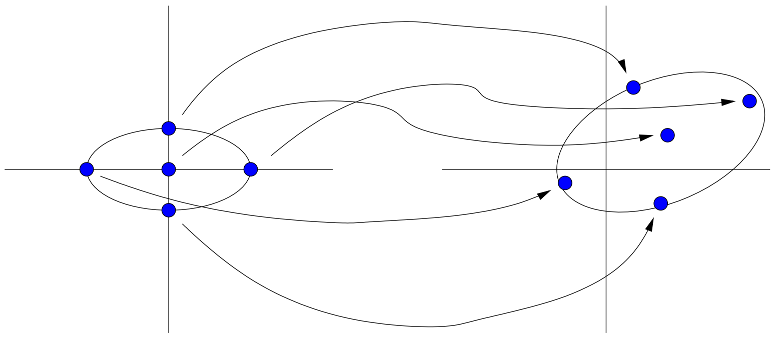


Figure 5.1: Unscented transform points are passed through the nonlinear model

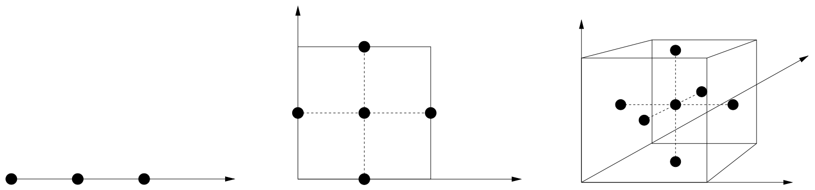


Figure 5.2: Distribution of unscented filter samples for 1, 2, and 3 dimensions.

$$\sum_i w_i(x_i - \mu)^5 \quad = 0 \tag{5.30}$$

Choosing the points to be symmetric ($x_0 = 0$, $x_1 = -x_2$ and $w_1 = w_2$) automatically satisfies the constraints on the mean, skew, and fifth moment. Using the criteria on normalized weights, variance and kurtosis criteria Julier et al. find that

$$\begin{aligned} \mathcal{X} &= \{0, -\sqrt{3}, \sqrt{3}\} \\ \mathcal{W} &= \left\{\frac{2}{3}, \frac{1}{6}, \frac{1}{6}\right\} \end{aligned} \tag{5.31}$$

are the best choice[44]. These samples correspond with the samples for the divided difference filter.

In higher dimensions, the Unscented filter repeats this 1-dimensional distribution along each dimension in the state space. The central point is used once, and one point is generated on either side of the central point along each dimension, so a total of $2d + 1$ points are needed. The weights are set to $\kappa/(d + \kappa)$ for the central point, and $1/(2(d + \kappa))$ for all other points, and a guideline is given that κ should be set such that $d + \kappa = 3$. For example, the five points and weights used for a 2D state space would be

$$\mathcal{X} \quad = \quad \left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} -\sqrt{3} \\ 0 \end{pmatrix}, \begin{pmatrix} \sqrt{3} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -\sqrt{3} \end{pmatrix} \begin{pmatrix} 0 \\ \sqrt{3} \end{pmatrix} \right\}$$

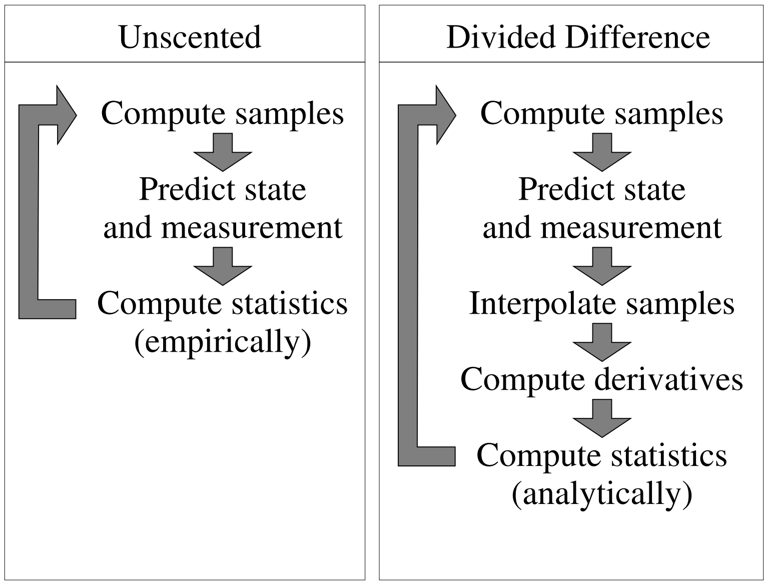

Unscented
Compute samples
Predict state and measurement
Compute statistics (empirically)
Divided Difference
Compute samples
Predict state and measurement
Interpolate samples
Compute derivatives
Compute statistics (analytically)

Figure 5.3: Flow diagram of the Unscented and Divided Difference filters

$$\mathcal{W} = \left\{\frac{1}{3}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}\right\} \tag{5.32}$$

The arrangement of Unscented filter points are shown for 1, 2, and 3 dimensions in Figure 5.2.

The sigma points in the unscented filter are passed through the process or observation model and the mean and covariance of the resulting points is computed. A schematic of this step is shown in Figure 5.1. Julier points out that in Jacobian based methods, while the state estimate vector is passed through the process or observation model, the errors pass through a separate linear system[44]. The sigma points of the unscented filter pass through the same nonlinear function as the state vector. Some loss of information occurs when the posterior mean and variance are computed, but the resulting mean and variance should be closer to the true values.

Julier and Uhlmann later introduced a scaling parameter which allows a filter designer to match higher order moments in the distribution at the expense of no longer matching some of the lower order moments as well. The resulting transform is referred to as the *scaled unscented transform*[46].

Julier has also introduced a "simplex" form of the unscented transform, using the minimum number of points required to match the first two moments[45]. The points now form a simplex consisting of $N+1$ vertices in N dimensions.

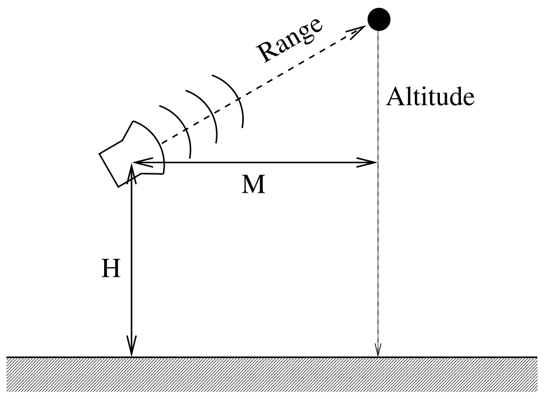


Figure 5.4: Ballistic reentry problem from Athans (1968)

5.3.3 Relation between UKF and DD2

The UKF and DD2 filters are actually quite similar. The Unscented filter samples a set of points which exactly represent the mean and covariance of the prior distribution, evaluate the state prediction and measurement prediction for each sample, and compute the posterior mean and covariance empirically using the sample points. The DD2 filter samples a set of points which exactly represent the mean and covariance, predicts the state and measurement for each sample, finds a second order polynomial which best interpolates the predictions, computes derivatives of the interpolated function, and uses the derivatives to compute the posterior mean and covariance analytically. The difference between the two methods can is shown in Figure 5.3.

The sampling scheme is also shared between the two filters. Each filter approximates the 1-D standard normal with the points $\mathcal{X} = \{-\sqrt{3}, 0, \sqrt{3}\}$ and weights $\mathcal{W} = \{\frac{1}{6}, \frac{2}{3}, \frac{1}{6}\}$, and approximates a general d dimensional Gaussian by repeating this rule along each principal direction and scaling by the variance.

In order to demonstrate the similar performance of these two algorithms, they have been applied to a standard benchmark problem from a 1968 paper by Athans et al.[3]. The paper presents a ballistic reentry problem which has been widely used as a benchmark for nonlinear estimation. Julier and Nørgaard have provided Matlab implementations of the UKF[91] and DD2[67] filters, respectively, including the Athans ballistic reentry benchmark problem.

A diagram of the simulated ballistic reentry problem is in Figure 5.4. A ballistic body with known altitude and velocity enters the atmosphere. A radar repeatedly measures the range to the object as it falls. Gravity accelerates the body downward, but drag causes the object to decelerate as it enters the atmosphere. Because the filter begins with the wrong ballistic coefficient, the filter makes biased predictions about the deceleration due to air drag. The benchmark problem is designed to see how fast and how well the filter can recover from the bias.

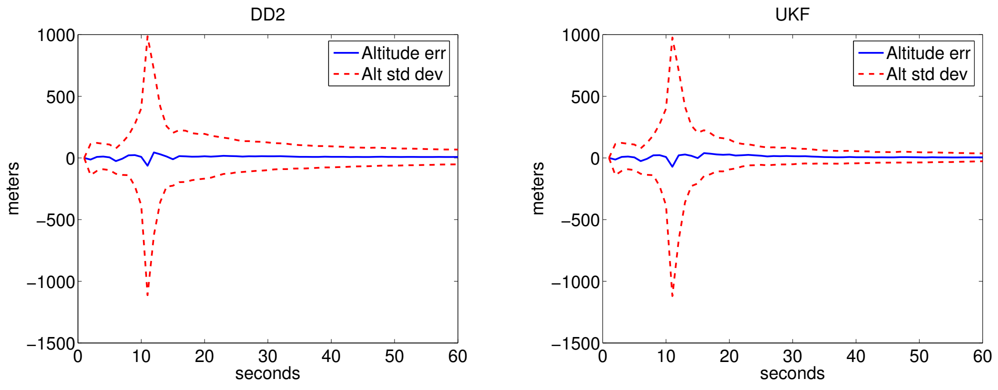


Figure 5.5: Altitude estimation errors for DD2 and Unscented filter.

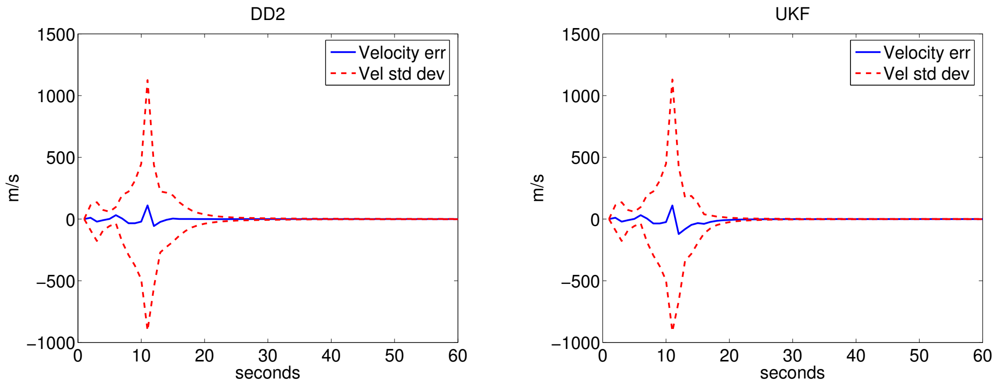
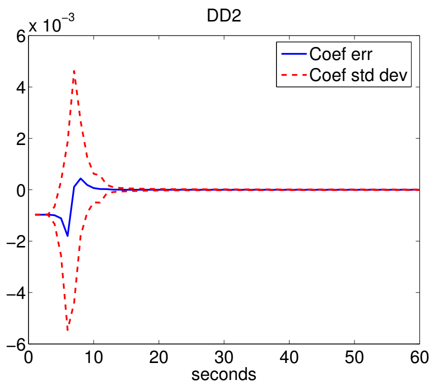


Figure 5.6: Velocity estimation errors for DD2 and Unscented filter.

Results from applying the UKF and DD2 to the problem are in Figure 5.5 through 5.9. The unscented filter does appear to outperform the DD2 filter, but not in a significant way compared to the improvement that both filters achieve beyond the EKF. This is not surprising since the deterministic sampling scheme in each filter is the same. What is interesting to note, however, is that the two filters are derived using different principles and result in very similar performance.



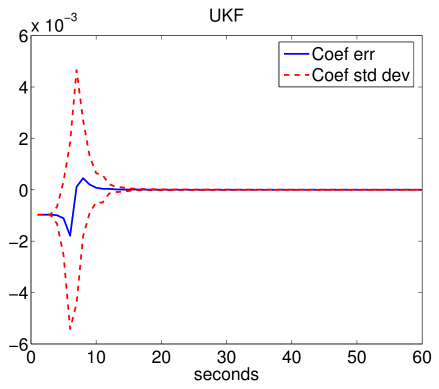
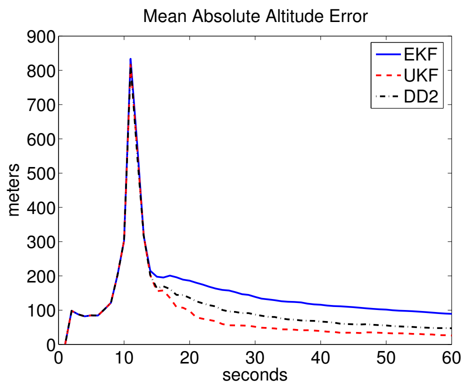
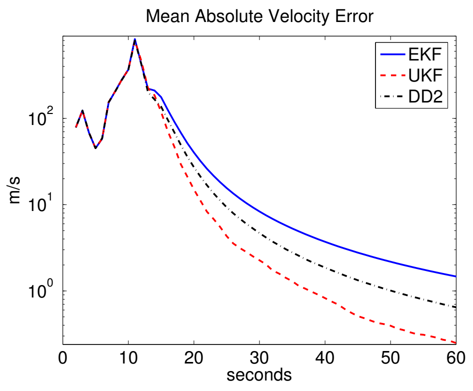


Figure 5.7: Ballistic coefficient estimates for DD2 and Unscented filter.



(a)



(b)

Figure 5.8: Average absolute error for the Unscented and DD2 filters. (a) Altitude. (b) Velocity.

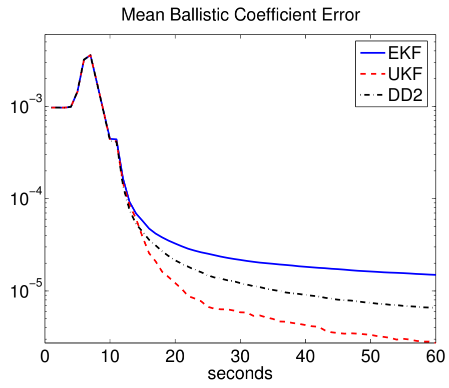


Figure 5.9: Average absolute error in ballistic coefficient for the Unscented and DD2 filters

5.3.4 A "more robust" unscented filter

The unscented transform (and filter) was first introduced in this form in 1995[44]. van Zandt later introduced a more robust method using a larger number of samples along each axis[94]. Unfortunately the word robust is not used here in the sense that is typically is in statistics, where the influence of outliers is of concern. Rather the method is supposed to provide more accurate estimates of posterior statistics in the presence of strong nonlinearities or discontinuities.

The points in van Zandt's robust unscented filter are taken from the roots of higher order orthogonal polynomials, or equivalently from higher order Gauss-Hermite quadrature rules in one dimension. The examples he shows use an odd number of points so that there is one central point. In higher dimensions, van Zandt's filter embeds $Nd+1$ points in the same way that Julier and Uhlmann's original filter uses $2d+1$, i.e. by repeating 1-dimensional rules along the directions given by the columns of the Cholesky factor of $\widehat{C}$.

If the system or measurement model is highly nonlinear or discontinuous, then larger samples will generate answers closer to the true posterior mean and variance. Of course if the system and measurement are truly discontinuous then an approach which can handle a multimodal, nonGaussian posterior might be preferred, such as particle filters.

5.3.5 Quadrature based method

Gaussian quadrature is a means of numerically computing an integral using a small number of carefully chosen points and associated weights[71]. Deterministic rules for computing the samples and weights exist. There are specific rules for computing the samples to use for evaluating expectations as in (5.3) depending on the form of the distribution over which the expectation is computed.

In place of the unscented transform and unscented filter, Ito et al. compute the integrals in (5.7) and (5.9) using Gaussian Quadrature[43]. The algorithm

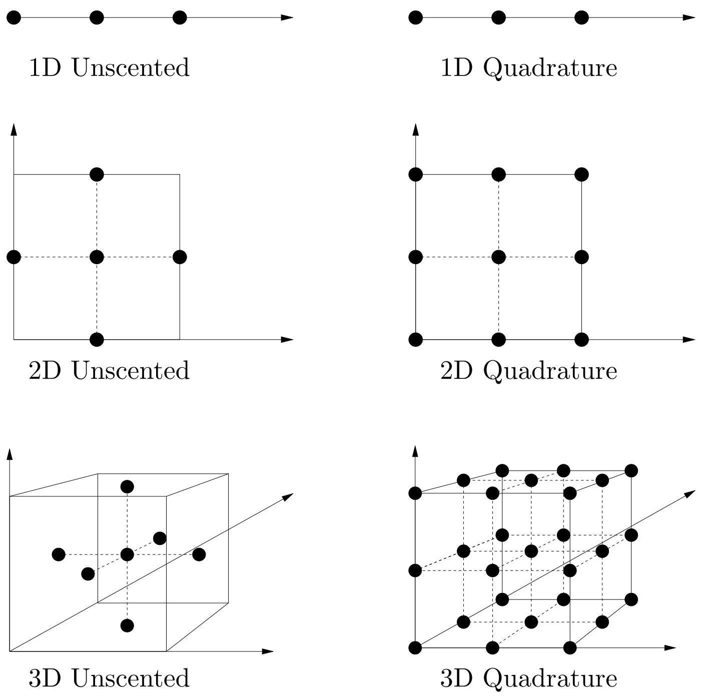


Figure 5.10: Comparison of sample points for unscented and quadrature methods in 1, 2, and 3 dimensions.

proceeds in the same way as the unscented filter.

Figure 5.10 shows the difference between the sigma points from Unscented filtering and the quadrature points for a one, two and three dimensional standard normal. The important distinction is in the "off diagonal" points. The sigma points lie along the axes, which one can visualize as the line segments that connect faces of a hypercube. There are $2d+1$ sigma points for a d-dimensional space. In contrast, a 3 point quadrature rule extended into d dimensions has a total of 3^d points. This should be acceptable only when the individual measurements rely on a few dimensions of the state space. Ito et al. report using up to six dimensional state spaces[43], which is far too few for a SLAM problem with even a few landmarks and robot poses.

To make the problem with exponential numbers of samples more explicit, consider the minimal constraints on the number of robot poses and landmarks for which a solution to the SLAM problem exists. A minimal well posed problem is one with three robot poses and five landmarks. In a 2-D world, this means a 19 dimensional state space. Integrals defined over these 19 dimensions and using a 3-point quadrature rule recursively for each dimension would require $3^{19} = 1,162,261,467$, over one billion. A reasonable number of landmarks, on the order of 100 for example, would require in the neighborhood of 10^{100} samples.

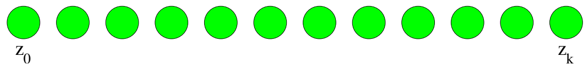


Figure 5.11: Linearization scheme for batch methods

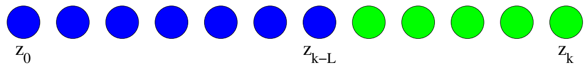


Figure 5.12: Linearization scheme for smoothing

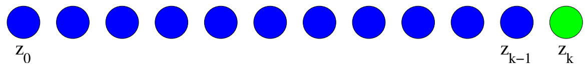


Figure 5.13: Linearization scheme for Kalman filters

5.4 Recursive nonlinear smoothing

Nonlinear smoothing retains states and measurements beyond the time step when they are first introduced. The algorithm estimates the state of the system over a specified time window rather than at a specific snapshot in time. Smoothing offers a trade-off between the one-step prediction and updating approach of filtering and full batch estimation methods. A schematic representation of this is in Figure 5.12. The green dots indicate measurements which are treated as nonlinear quantities, whereas the blue dots indicate measurements which have been linearized and approximated. Batch methods recompute local linearizations until the algorithm converges. Kalman filters compute a linearization right when the measurement is incorporated and then not reconsidered. Smoothing filters treat the measurements as nonlinear for some time lag, which may be fixed or determined on the fly, and linearize the measurements after they have been used to help determine the state of the system into the future.

There are many advantages to recursive smoothing over recursive filtering. Using information from multiple time steps helps to resolve degeneracies. One bearing measurement is not enough to initialize a new feature, but bearing measurements from two or more different locations can be used to triangulate the location of a new feature. The multiple constraints can also help to disambiguate the rover pose, helping to improve the motion estimate over one-step filtering of odometry.

Since the measurements will be linearized in the smoothing filter, the use of information from multiple time steps also helps to refine the state estimate before computing the linearization. This means that the linearization will be computed using an estimate which is more likely to be near the true solution, reducing linearization errors and better approximating the posterior covariance relative to the Fisher information at the true solution.

Smoothing also allows the algorithm to reconsider data association. This is the primary reason smoothing is used in the delayed initialization work of

Leonard and Rikoski[48].

Finally, smoothing allows us to develop a robust version of the recursive algorithm by incorporating influence functions from iteratively reweighted least squares (IRLS) in robust estimation.

5.4.1 Variable state dimension filter

The variable state dimension filter (VSDF) combines aspects of the EKF with aspects of Gauss-Newton nonlinear optimization[71]. Since $\mathbf{z_k}$ is a Gaussian random variable with mean $\mathbf{h_k}(\mathbf{x})$ and variance $\mathbf{R_k}$, we can write the likelihood for $\mathbf{z}$ given $\mathbf{x}$

$$p(\mathbf{z}|\mathbf{x}) \propto e^{-\sum_k (\mathbf{z_k}-\mathbf{h_k}(\mathbf{x}))^T \mathbf{R_k}^{-1}(\mathbf{z_k}-\mathbf{h_k}(\mathbf{x}))} \tag{5.33}$$

Gauss-Newton optimization searches for the parameter which minimizes the negative log of the likelihood

$$\begin{aligned} e &= -log(p(\mathbf{z}|\mathbf{x})) \\ &= \sum_k (\mathbf{z_k} - \mathbf{h_k}(\mathbf{x}))^T \mathbf{R_k}^{-1} (\mathbf{z_k} - \mathbf{h_k}(\mathbf{x})) \end{aligned} \tag{5.34}$$

In order to minimize this cost function, the algorithm starts with an estimate of the state vector $\hat{\mathbf{x}}_0$ and computes

$$\mathbf{a} = \sum_k -\mathbf{H_k}^T \mathbf{R_k}^{-1} (\mathbf{z_k} - \mathbf{h_k}(\mathbf{x})) \tag{5.35}$$

$$\mathbf{A} = \sum_k \mathbf{H_k}^T \mathbf{R_k}^{-1} \mathbf{H_k} \tag{5.36}$$

where $\mathbf{H_k} = \frac{\partial \mathbf{h_k}}{\partial \mathbf{x}}\big|_{\hat{\mathbf{x}}_0}$ is the measurement Jacobian, $\mathbf{a} = \nabla e$ is the gradient of (5.34) and $\mathbf{A} \approx \nabla^2 e$ is an approximation to the Hessian[71]. The algorithm computes an update to the state estimate by solving the linear system

$$\mathbf{A}\delta = \mathbf{a} \tag{5.37}$$

and updating the parameter vector

$$\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}} - \delta \tag{5.38}$$

Equations (5.35) through (5.38) are iterated to convergence. Solutions found using Gauss-Newton are optimal in a least squares sense, which is also maximum likelihood for Gaussian noise. However, the vector $\mathbf{x}$ contains the entire map and the entire vehicle trajectory, which makes Gauss-Newton slow for large datasets.

The VSDF provides a method for linearizing measurements, incorporating them into a Gaussian "prior". The filter equations may be derived by linearizing terms on the right hand side of (5.35) and (5.36). Suppose we wish to replace the term involving $\mathbf{z_k}$ in (5.34). We can compute a linear approximation to $\mathbf{h_k}()$

$$\mathbf{h_k}(\mathbf{x}) \approx \mathbf{h_k}(\mathbf{x}_0) + \mathbf{H_k}(\mathbf{x} - \mathbf{x}_0) \tag{5.39}$$

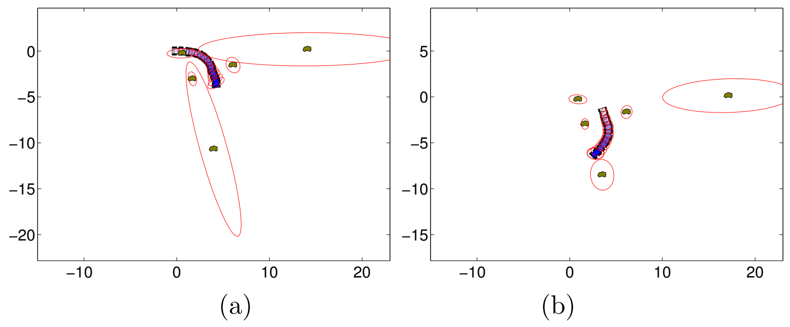


Figure 5.14: The variance of the state estimate decreases over time, making linearizations more accurate

and in order to minimize the new cost function we simply replace the corresponding term in (5.35) and (5.36) with the same linearization.

In the VSDF, terms are replaced with a linearization of the form

$$(\mathbf{x} - \hat{\mathbf{x}}_0)^T \mathbf{A_k} (\mathbf{x} - \hat{\mathbf{x}}_0) \approx (\mathbf{z_k} - \mathbf{h_k}(\mathbf{x}))^T R_k^{-1} (\mathbf{z_k} - \mathbf{h_k}(\mathbf{x})) \tag{5.40}$$

$$\begin{aligned}&(\mathbf{x} - \hat{\mathbf{x}}_0)^T \mathbf{A_k} (\mathbf{x} - \hat{\mathbf{x}}_0) \approx \\ &(\mathbf{z_k} - \mathbf{H_k}(\mathbf{x} - \hat{\mathbf{x}}_0))^T R_k^{-1} (\mathbf{z_k} - \mathbf{H_k}(\mathbf{x} - \hat{\mathbf{x}}_0))\end{aligned} \tag{5.41}$$

where $\mathbf{A_k} = \mathbf{H_k}^T R_k^{-1} \mathbf{H_k}$ is the contribution of measurement k to the Hessian, and

$$\mathbf{a_k} = \mathbf{H_k}^T R_k^{-1} (\mathbf{z_k} - \mathbf{h_k}(\mathbf{x}_0)) \tag{5.42}$$

is the constant contribution of measurement k to the gradient (5.35). In the original VSDF the linearization $\mathbf{H_k}$ is again taken to be the Jacobian of the measurement function evaluated at the state estimate.

This is similar to the EKF, except that the EKF linearizes each measurement immediately upon incorporation into the state estimate. The VSDF opens the possibility of linearizing the term at some later time[57]. The advantage in linearizing the measurement later is that the point of expansion for the linearization is estimated using more data, and therefore has smaller variance. We can expect the linearization to occur at a more accurately estimated point, as shown in Figure 5.14.

The error and its Jacobian and Hessian can now be expressed as a combination of terms from the linearized measurements and the nonlinear measurements

$$\begin{aligned}
e &= (\mathbf{x} - \mathbf{x}_0)^T \mathbf{A_0} (\mathbf{x} - \mathbf{x}_0) + \sum (\mathbf{z_k} - \mathbf{h_k}(\mathbf{x}))^T \mathbf{R_k}^{-1} (\mathbf{z_k} - \mathbf{h_k}(\mathbf{x})) \\
\mathbf{a} &= \mathbf{a_0} + \mathbf{A_0}(\mathbf{x} - \mathbf{x}_0) + \sum \mathbf{H_k}^T \mathbf{R_k}^{-1} (\mathbf{z_k} - \mathbf{h_k}(\mathbf{x})) \\
\mathbf{A} &= \mathbf{A_0} + \sum \mathbf{H_k}^T \mathbf{R_k}^{-1} \mathbf{H_k}
\end{aligned} \tag{5.43}$$

Once measurements are linearized, there are parts of the state space that will no longer be a part of new measurements coming in (like old robot poses). Those subspaces can be eliminated from the filter. If we partition the state vector into

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} \tag{5.44}$$

and the gradient

$$\mathbf{a} = \begin{bmatrix} \mathbf{a_1} \\ \mathbf{a_2} \end{bmatrix} \tag{5.45}$$

and

$$\mathbf{A} = \begin{pmatrix} \mathbf{A_{11}} & \mathbf{A_{12}} \\ \mathbf{A_{21}} & \mathbf{A_{22}} \end{pmatrix} \tag{5.46}$$

then we can eliminate the first subspace by updating the parameter vector, gradient, and Hessian as follows,

$$\begin{aligned} \mathbf{x} &\leftarrow \mathbf{x}_2, \quad \mathbf{a} \leftarrow \mathbf{a_2} \\ \mathbf{A} &\leftarrow \mathbf{A_{22}} - \mathbf{A_{21}}\mathbf{A_{11}}^{-1}\mathbf{A_{12}} \end{aligned} \tag{5.47}$$

See [57] for details. The filter continues to incorporate new measurements as they become available, and linearizes them at some later time.

5.5 Conclusions

Recursive state estimation is attractive because of the online constraints faced by real world robotics applications. Batch methods are infeasible for long duration robotic tasks since the amount of data collected can, for all practical purposes, be considered unbounded. Even in hierarchical mapping approaches, a method is required for building local submaps, and a recursive filter is a good candidate for the task if submaps are to be built with large amounts of information.

Kalman filtering based approaches have dominated the SLAM community for over a decade, but there are issues with bias caused by incorrect linearization of nonlinear process and measurement models. These problems are exacerbated in a bearing only SLAM context since no range information is available from the sensor, and the problems is at its worst for initialization of new features when few measurements are available to disambiguate the location of the feature even in local rover coordinates. For this reason an approach which can consider multiple robot poses in a smoothing context is desired.

In terms of computing posterior statistics, the unscented filter is a great improvement over the traditional Kalman filter for several reasons. The filter computes posterior statistics which are more accurate than the Jacobian based method because the linearization is done by considering a neighborhood on a length scale which is commensurate with the uncertainty in the filter. Also, the tedious task of computing analytic Jacobians and encoding them in a Kalman filter can be avoided. However, as it stands, the unscented filter only uses one step prediction to compute the posterior statistics. There is currently no adaptation to the filter which allows computation of the posterior statistics using the updated state rather than the predicted state, as is done in the IEKF. Linearizing about the posterior mean rather than the predicted mean can improve Kalman filtering and should therefore improve unscented filtering. In addition, the extension of the unscented filter for multiple dimensions is incomplete. The unscented filter points correspond to an optimal Gaussian quadrature rule in one dimension, but the multi-dimensional extension simply repeats the one-dimensional integral over each dimension. Finally, there is currently no method for incorporating future data in a smoothing context to further improve the state estimate and accompanying linearization. The variable state dimension filter uses nonlinear smoothing to improve state estimation but uses a Jacobian based linearization. Since a deterministic sampling scheme has been shown to outperform the Jacobian in a one-step Kalman filter, we can expect to improve nonlinear smoothing by computing better statistics.

Chapter 6

Nonlinear smoothing and statistical linearization

This chapter describes the proposed recursive nonlinear smoothing method for bearing only localization and mapping. The method developed here relies on the development of the variable state dimension filter (VSDF)[57] and ideas from derivative free state estimation[44, 68] and robust statistics[42]. Using a nonlinear smoothing framework, we first investigate the optimal computation of linearized process and measurement models by considering statistical linearization, which attempts to find a linearization which is most accurate with respect to the state estimate and uncertainty. Closed form computation of the linearizations proves infeasible, but we can use concepts from exact numerical integration[71] and approximate Monte Carlo integration[51] to solve the problem. We finally incorporate a robust cost function[42] into nonlinear smoothing in order to achieve reasonable behavior in the presence of outliers.

6.1 Smoothing

Smoothing refers to the estimation of the state of a system at time i using observations from time $i + L$, i.e. in the future. There are many advantages to recursive smoothing over recursive filtering. Using information from multiple time steps helps to resolve degeneracies. One bearing measurement is not enough to disambiguate the position of a feature in order to initialize its location in the filter. Bearing measurements from two or more different locations can be used to triangulate the location of a new feature to improve initialization. The multiple constraints can also help to disambiguate the rover pose, helping to improve the motion estimate over one-step filtering. When measurements are linearized, they are done so after the state has been estimated using more data. This decreases the variance on the state estimate before linearization, so that the expansion point used has a better chance of being close to the true location. The inclusion of more information also helps to produce a posterior density which is more Gaussian, so that a Gaussian approximation to the posterior fits better. Smoothing also allows the filter to reconsider data association. Until a measurement is linearized and then discarded, it can be re-associated with a different feature. This feature is not used in the work presented here but it might become an important feature in future work. Finally, smoothing allows us to more easily include a robust loss loss function.

The extended Kalman filter (EKF) is not sufficient for bearing-only or range-only SLAM. The primary issue with the EKF approach is in initializing new features. One bearing or range measurement is not sufficient for disambiguating where a feature is in the environment relative to the rover. A bearing measurement restricts a feature to lie along a ray, but without range information the location along the ray is unknown. Because the location along the ray is unknown, any point along the ray is equally valid as an expansion point for linearization. However, using the wrong point for linearization can cause problems. With multiple bearing measurements made from different locations, the rover trajectory and feature positions can be uniquely recovered at the same time. With at least three poses and five landmarks, or at least four poses and four landmarks, the bearing measurements are enough to disambiguate the motion of the robot and the locations of the features up to scale, so the bearing measurements can compensate for errors in odometry if multiple time steps are considered. Performing smoothing in a consistent way requires using multiple measurements and treating them as the nonlinear quantities that they are rather than linearizing them as they are observed.

The smoothing algorithm presented here relies on the EKF and variable state dimension filter (VSDF) framework. The posterior density over state space is approximated by a Gaussian, parameterized by a mean and covariance which capture everything there is to know if the posterior is Gaussian. Measurements are linearized and removed from the filter in order to compute the Gaussian approximation. The primary issue is where, when, and how to compute the linearization.

6.2 Approximating the posterior

For linear process and measurement models, the posterior statistics are trivial to compute. If the measurement function is linear

$$h(\mathbf{x}) = h_0 + \mathbf{H}\mathbf{x} + \nu \tag{6.1}$$

where $\nu \sim \mathcal{N}(0, R)$ then the EKF update equations

$$\begin{aligned} \widehat{\mathbf{C}}_{t|t}^{-1} &= \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} + \widehat{\mathbf{C}}_{t|t-1}^{-1} \\ \hat{\mathbf{x}}_{t|t} &= \hat{\mathbf{x}}_{t|t-1} + \widehat{\mathbf{C}}_{t|t}\mathbf{H}_t^T\mathbf{R}_t^{-1}(z_t - h(\mathbf{x}_{t|t-1})) \end{aligned} \tag{6.2}$$

are exact in the sense that $\mathcal{N}(\hat{\mathbf{x}}_{t|t}, \widehat{\mathbf{C}}_{t|t})$ is the exact posterior distribution for the state estimate. However, for nonlinear functions the matrix $\mathbf{H}$ represents some linear approximation to the function $h(\mathbf{x})$. The EKF uses a Taylor series expansion of the measurement function

$$h(\mathbf{x}) = h(\mathbf{x}_0) + \mathbf{H}(\mathbf{x} - \mathbf{x}_0) + \frac{1}{2}\frac{\partial^2 h}{\partial \mathbf{x}^2}(\mathbf{x} - \mathbf{x}_0)^2 \ldots \tag{6.3}$$

where $\mathbf{H} = \frac{\partial h}{\partial \mathbf{x}}$, and then truncates the series after the linear term. This approximation is accurate for small departures $(\mathbf{x} - \mathbf{x}_0)$. A better approximation can be had by finding the linear function which best matches the nonlinear model over some neighborhood specified by the uncertainty in the state estimate. If the uncertainty in the state estimate is large, then the filter should use an approximation which is reasonable for large values of $(\mathbf{x} - \mathbf{x}_0)$. If the uncertainty is small then an approximation which better captures the shape of $h()$ for small values of $(\mathbf{x} - \mathbf{x}_0)$ is desirable.

The nonlinear smoothing algorithm approximates the true full posterior by approximating measurements with Gaussians and incorporating them into the prior term. At any time, the filter contains a Gaussian prior

$$p(\mathbf{x}) \propto \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}_0)^T \mathbf{A}_0 (\mathbf{x} - \mathbf{x}_0)\right) \tag{6.4}$$

and a true measurement likelihood

$$p(\mathbf{z}|\mathbf{x}) \propto \prod_k \exp\left(-\frac{1}{2}(z_k - h_k(\mathbf{x}))R_k^{-1}(z_k - h_k(\mathbf{x}))\right) \tag{6.5}$$

where each z_k is a one dimensional measurement (single bearing to a landmark). The posterior $p(\mathbf{x}|\mathbf{z})$ is proportional to the product of (6.4) and (6.5). Maximizing the posterior is done by computing the negative log of the posterior to obtain the objective function

$$J = (\mathbf{x} - \mathbf{x}_0)^T \mathbf{A}_0 (\mathbf{x} - \mathbf{x}_0) + \sum_k (z_k - h_k(\mathbf{x}))R_k^{-1}(z_k - h_k(\mathbf{x})) \tag{6.6}$$

When measurement z_j is to be moved from the nonlinear portion of the filter (the sum in (6.6)) to the linearized portion (the first quadratic term, or prior, in (6.6)), the approximation

$$(\mathbf{h}_1^T \mathbf{x} - h_0)R_j^{-1}(\mathbf{h}_1^T \mathbf{x} - h_0) \approx (z_j - h_j(\mathbf{x}))R_j^{-1}(z_j - h_j(\mathbf{x})) \tag{6.7}$$

is required, where the constant scalar h_0 and the $N \times 1$ coefficient vector $\mathbf{h}_1$ form the linearization of the term involving the observation z_j and measurement function $h_j(\mathbf{x})$. As noted earlier, the standard EKF implementation uses the measurement Jacobian for $\mathbf{h}_1$ and the value of the measurement at the nominal state estimate for h_0. However, in order to minimize the effect of this approximation on the cost function, one can instead compute a linearization for $h_j(\mathbf{x})$ which minimizes the expected squared error between the approximation and the true function

$$e = \int p(\mathbf{x}|\mathbf{z})(z_j - h_j(\mathbf{x}) - \mathbf{h}_1^T \mathbf{x} + h_0)R_j^{-1}(z_j - h_j(\mathbf{x}) - \mathbf{h}_1^T \mathbf{x} + h_0)d\mathbf{x} \tag{6.8}$$

with respect to the linearization parameters $\mathbf{h}_1$ and h_0. This is the statistical linearization[54] of the function $\mathbf{h}(\mathbf{x})$, and corresponds to the minimum expected future squared deviation between the nonlinear model and the linear approximation, with the expectation taken under the current posterior over state space.

Using the notation $E_{p(\mathbf{x}|\mathbf{z})}\{\}$ to denote expectation, we write (6.8) as

$$e = E_{p(\mathbf{x}|\mathbf{z})}\{(z_j - h_j(\mathbf{x}) - \mathbf{h}_1^T\mathbf{x} + h_0)R_j^{-1}(z_j - h_j(\mathbf{x}) - \mathbf{h}_1^T\mathbf{x} + h_0)\} \tag{6.9}$$

The minimum of (6.9) with respect to $\mathbf{h}_1$ and h_0 occurs where the partial derivatives of $e()$ with respect to those parameters vanish,

$$\begin{aligned} \frac{\partial e}{\partial \mathbf{h}_1} &= E_{p(\mathbf{x}|\mathbf{z})}\{-2\mathbf{x}R_j^{-1}(z_j - h_j(\mathbf{x}) - \mathbf{h}_1^T\mathbf{x} + h_0)\} \\ \frac{\partial e}{\partial h_0} &= E_{p(\mathbf{x}|\mathbf{z})}\{2R_j^{-1}(z_j - h_j(\mathbf{x}) - \mathbf{h}_1^T\mathbf{x} + h_0)\} \end{aligned} \tag{6.10}$$

Setting the partial derivatives to zero, multiplying by the constant $R_j/2$, separating the terms in the expectation, and taking advantage of the fact that the inner product $\mathbf{h}_1^T\mathbf{x} = \mathbf{x}^T\mathbf{h}_1$ leads to

$$\begin{aligned} \left(E_{p(\mathbf{x}|\mathbf{z})}\{\mathbf{x}\mathbf{x}^T\}\right)\mathbf{h}_1 - \left(E_{p(\mathbf{x}|\mathbf{z})}\{\mathbf{x}\}\right)h_0 &= E_{p(\mathbf{x}|\mathbf{z})}\{x(z_j - h_j(\mathbf{x}))\} \\ -\left(E_{p(\mathbf{x}|\mathbf{z})}\{\mathbf{x}^T\}\right)\mathbf{h}_1 + h_0 &= -E_{p(\mathbf{x}|\mathbf{z})}\{(z_j - h_j(\mathbf{x}))\} \end{aligned} \tag{6.11}$$

which is a linear system of equations for $\mathbf{h}_1$ and h_0,

$$\begin{pmatrix} E_{p(\mathbf{x}|\mathbf{z})}\{\mathbf{x}\mathbf{x}^T\} & -E_{p(\mathbf{x}|\mathbf{z})}\{x\} \\ -E_{p(\mathbf{x}|\mathbf{z})}\{\mathbf{x}^T\} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{h}_1 \\ h_0 \end{pmatrix} = \begin{pmatrix} E_{p(\mathbf{x}|\mathbf{z})}\{\mathbf{x}(z_j - h_j(\mathbf{x}))\} \\ -E_{p(\mathbf{x}|\mathbf{z})}\{(z_j - h_j(\mathbf{x}))\} \end{pmatrix} \tag{6.12}$$

In order to solve (6.12), we must compute the integrals which form the coefficients in the linear system. Since $p(\mathbf{x}|\mathbf{z})$ is a complicated function, the integrals in (6.12) cannot be computed directly, but numerical integration techniques can be applied as discussed in the next section.

As an example of what statistical linearization does, Figure 6.1 shows the linearization of a 1-dimensional nonlinear function under a Gaussian density. The blue curve is the nonlinear function to be approximated. The purple dots show the Gaussian density over which the expectation values are computed. The dashed red line shows the Jacobian, or tangent plane, and the solid red line shows the statistical linearization, or the line which minimizes (6.8). As the mean of the Gaussian changes, a different part of the function is approximated. As the variance of the Gaussian increases, the neighborhood over which the function must be approximated grows. The Jacobian would correspond to the statistical linearization if the posterior $p(x|z)$ were a delta function, and would be a reasonable approximation if the Gaussian had an extremely low variance. However, for large variance the statistical linearization can be dramatically different. In the SLAM problem, errors can accumulate over time and over large distances, particularly when the same features are not always in view, and the

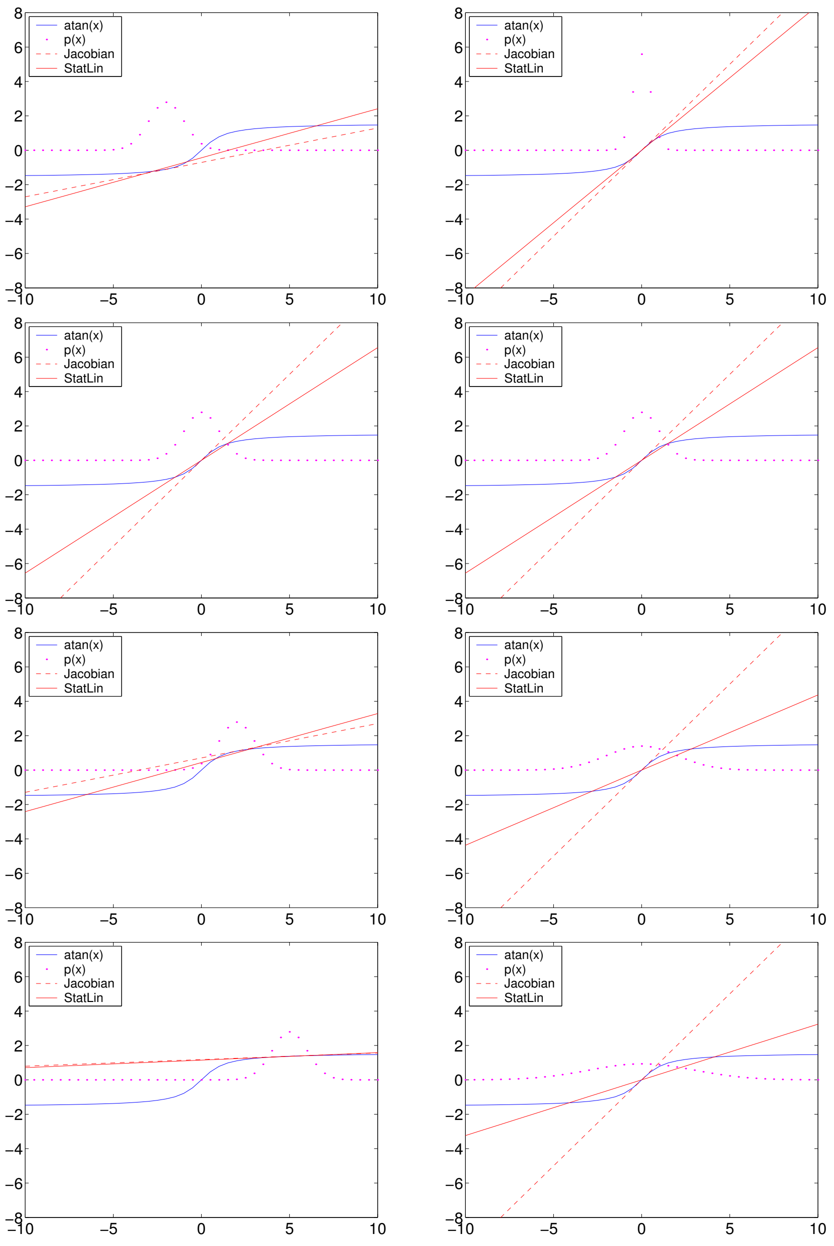


Figure 6.1: Comparison of Jacobian and statistical linearization for different mean and variance $p(x)$

variance of the posterior can become significant enough that the difference in linearizations is important.

The numerical technique used to integrate the terms in (6.12) is the subject of the next section.

6.3 Numerical integration

There are many integrals of interest which cannot be computed analytically. In these cases, numerical techniques for approximating integrals are required. Gaussian quadrature and Monte Carlo techniques have been used in recursive state estimation recently, and can be used to compute posterior statistics in recursive nonlinear smoothing.

6.3.1 Quadrature

The term *quadrature* refers to any numerical integration method which seeks to replace an integral with a sum. Newton-Cotes rules, including the familiar trapezoidal rule and Simpson's rule, use points which are equally spaced in the interval $[a, b]$, and weights such that the sum is exactly correct for some class of functions[71].

$$\int_a^b f(x)\,dx = \sum_i w_i f(x_i); \qquad x_i = a + i\frac{(b-a)}{N}; \quad i = 0 \dots N \tag{6.13}$$

For the two rules mentioned above, the classes of functions which can be integrated exactly are first and second order polynomials, i.e. linear and quadratic functions, respectively. Newton-Cotes rules specify the weights but not the points at which the function is evaluated.

If the function $f()$ is indeed linear, then the trapezoidal rule can compute the integral exactly using only two samples. This should come as no surprise since a line is specified uniquely by two points. The two samples should contain all of the information available about the line and therefore all the information available about its integral. If the function is quadratic, then Simpson's rule can compute the result exactly using only three samples for similar reasons. When the function is not linear or quadratic, the answer is only approximate.

6.3.2 Gaussian quadrature

Gaussian quadrature is another method for replacing integrals with sums. In addition to the weights for the sum, Gaussian quadrature rules specify the locations at which the function should be evaluated. This offers two degrees of freedom per sample in the sum, which gives more flexibility and therefore higher order approximations with the same number of samples. Furthermore, Gaussian quadrature rules are designed to compute integrals of polynomials multiplied by

Quadrature rule	Weight function	Interval
Gauss-Legendre	$w(x) = 1$	$-1 < x < 1$
Gauss-Chebychev	$w(x) = (1-x^2)^{-1/2}$	$-1 < x < 1$
Gauss-Leguerre	$w(x) = x^\alpha e^{-x}$	$0 < x < \infty$
Gauss-Hermite	$w(x) = e^{-x^2}$	$-\infty < x < \infty$
Gauss-Jacobi	$w(x) = (1-x)^\alpha (1+x)^\beta$	$-1 < x < 1$

Table 6.1: Gaussian quadrature rules for some weight functions, from[71].

some known weighting function, i.e.

$$\int_x w(x) f(x)\, dx = \sum_i w_i f(x_i) \tag{6.14}$$

The weighting function determines which Gaussian quadrature rule applies. Table 6.1, adapted from *Numerical Recipes*[71], has a list of weighting functions and corresponding exact quadrature rules. A Gaussian quadrature rule using n points can evaluate (6.14) exactly when the polynomial $f()$ is order $2n-1$ or less[71]. This means that if the polynomial is order d, the Gaussian quadrature rule requires $(d+1)/2$ samples in order to compute the result exactly. For functions which are not polynomial, the error in the integral will be related to the lowest order component of the function which cannot be captured by the quadrature rule.

Gauss-Hermite quadrature can compute integrals exactly for polynomial functions and a weighting of the form e^{-x^2}. The points and weights for a Gauss-Hermite quadrature rule with 3, 5, 7, 9, 11, and 13 points are shown in Figure 6.2.

To use a normal distribution as the weighting function $w(x) = \mathcal{N}(\mu, \sigma)$, we need to modify the samples. A change of variables using $u = \frac{x-\mu}{\sqrt{2}\sigma}$ shows the correction,

$$\begin{aligned}\int_x \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) f(x)\, dx &= \frac{1}{\sqrt{\pi}} \int_u \exp(-u^2) f(\sqrt{2}\sigma u + \mu)\, du \\ &= \frac{1}{\sqrt{\pi}} \sum_i w_i f(\sqrt{2}\sigma u_i + \mu) \qquad (6.15)\end{aligned}$$

To compute expectations with respect to $\mathcal{N}(\mu, \sigma)$, we must scale the abscissae by $\sqrt{2}\sigma$ and add μ, and multiply the weights by $\frac{1}{\sqrt{\pi}}$. In multiple dimensions, the points are modified by multiplying by $\sqrt{2}\mathbf{S}$ where $\mathbf{S}\mathbf{S}^T = \mathbf{C}$, adding the mean vector $\bar{\mathbf{x}}$, and multiplying the weights by $\frac{1}{\pi^{d/2}}$.

The integral of interest in computing posterior statistics for recursive filtering is an expectation taken under the posterior distribution $p(x|x_0, A_0, z)$. The posterior is conditioned on the mean state vector and covariance which were computed by linearizing older measurements, as well as the nonlinear measurements which are still in the filter. This distribution is approximately Gaussian,

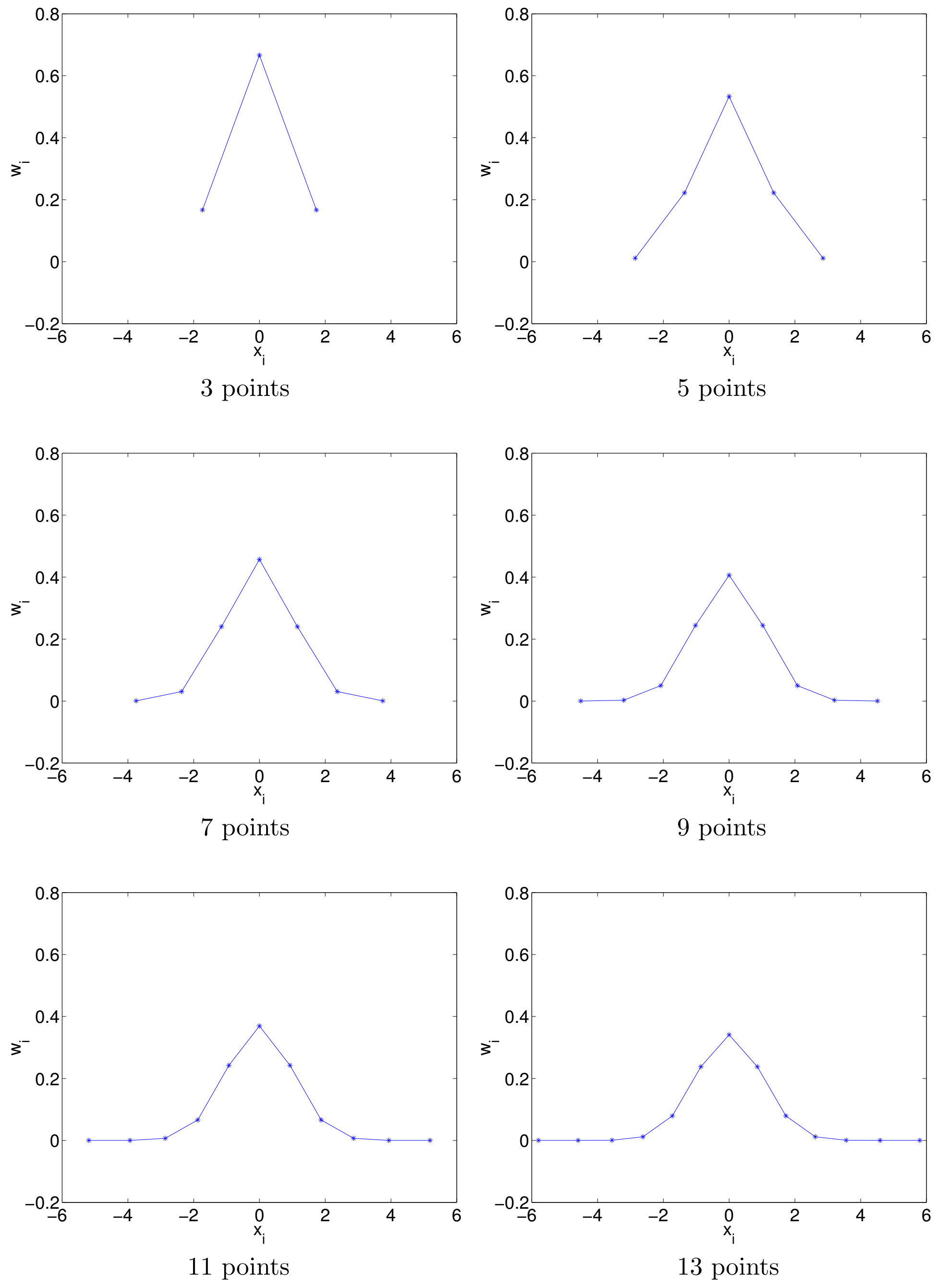


Figure 6.2: Gauss-Hermite samples and weights for 3, 5, 7, 9, 11, and 13 points

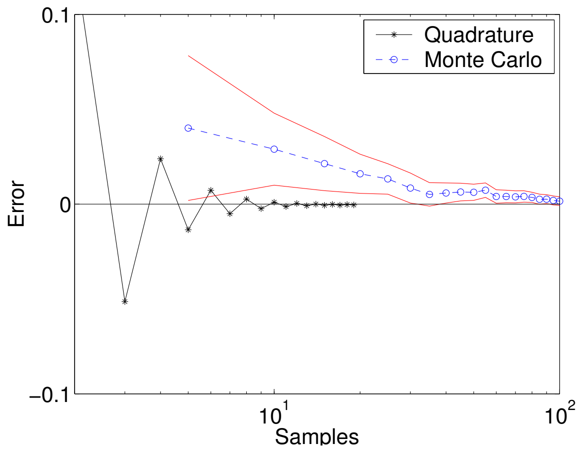


Figure 6.3: Integration errors for quadrature and Monte Carlo with varying sample sizes.

but not exactly Gaussian. We can take this into account by modifying the quadrature rule using a technique from Monte Carlo.

6.3.3 Monte Carlo

Monte Carlo methods are also a popular method for converting the integral in (6.14) to a sum. Monte Carlo techniques can compute arbitrary integrals to arbitrary precision given enough computation, but convergence only happens in the limit as the number of samples goes to infinity. The techniques are widely used since not all distributions fit the rules in Table 6.1 and not all functions of interest are limited order polynomials.

If $w(x)$ is a probability distribution from which we can sample $x_i \sim w(x)$, then $w_i = 1$ for all samples and the integral is approximated by

$$\int w(x) f(x)\, dx \approx \sum_i f(x_i); \qquad x_i \sim w(x) \tag{6.16}$$

One important difference between quadrature rules and Monte Carlo is the bias and variance in the estimates. Quadrature rules are biased, unless the function being integrated is a bounded order polynomial in which case the integral is exact. Quadrature is always zero variance; the samples are computed deterministically. Monte Carlo, on the other hand, is statistically unbiased but has finite variance. The variance decreases as the sample size increases. Figure 6.3 shows the error in computing the linearization for $f(x) = \arctan(x)$ and $p(x) = \mathcal{N}(0,1)$ using Gauss-Hermite quadrature and using Monte Carlo. The quadrature rule converges quickly to the correct answer. The Monte Carlo method has considerable variance for small sample sizes, and even after sample

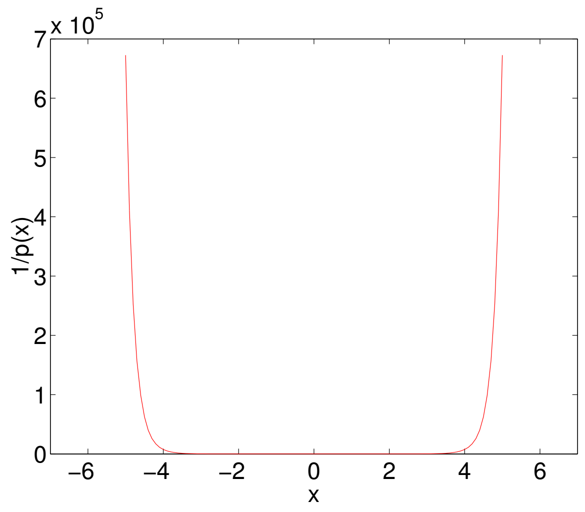


Figure 6.4: Ratio $1/q(x)$ for a standard normal proposal density.

sizes of over 100 has not yet reached an average error less than a quadrature rule using 11 points.

6.3.4 Importance sampling

Not all probability densities are easy to sample from. If the probability density $p(x)$ cannot be directly sampled, but can be evaluated (even up to a constant scale factor) then *importance sampling* can be used. A *proposal distribution* $q(x)$ is used to draw samples, and *importance weights* are computed as a ratio between the true distribution and the proposal distribution[51] to compute the integral. That is,

$$\begin{aligned} \int p(x)f(x)\,dx &= \int q(x)\frac{p(x)}{q(x)}f(x)\,dx \\ &\approx \sum_i \frac{p(x_i)}{q(x_i)}f(x_i); \qquad x_i \sim q(x) \end{aligned} \tag{6.17}$$

There are some caveats to using importance sampling, which are discussed in an excellent tutorial on Monte Carlo by MacKay[51]. Of principal interest is the caveat that $q(x)$ must have support wherever $p(x)$ is nonzero, or the weight $p(x)/q(x)$ can diverge. One sample from a region of low probability in $q(x)$ can dominate the sum if the weight is large. Figure 6.4 shows $1/q(x)$ for a standard normal proposal $q(x) = \mathcal{N}(0, 1)$. This quantity is multiplied by the value of the true density to get the importance weight. The divergence of $1/q(x)$ for large values of x is dangerous if $p(x)$ has support where the divergence occurs.

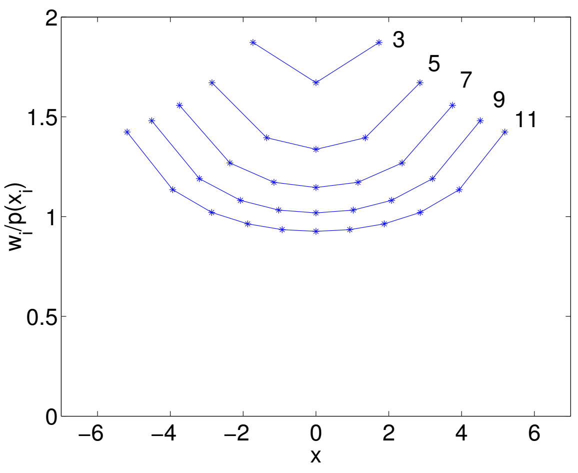


Figure 6.5: Ratio $w_i/q(x_i)$ for a standard normal proposal density and Gauss-Hermite quadrature with 3, 5, 7, 9, and 11 points.

6.3.5 Importance quadrature

If the integral we wish to compute is of the form

$$\int p(x)f(x)\,dx \approx \sum w_i f(x_i) \tag{6.18}$$

and $p(x)$ is a normal distribution, then we can apply the Gauss-Hermite quadrature rule as described above. However, if the distribution $p(x)$ is not normal, then Gauss-Hermite quadrature does not strictly apply. If $p(x)$ can be reasonably approximated by a normal density $q(x)$, then we can use the importance weighting idea from Monte Carlo to compute (6.18). Using $q(x) = \mathcal{N}(\mu, \sigma)$ as a proposal density, (6.18) becomes

$$\int q(x)\frac{p(x)}{q(x)}f(x)\,dx \approx \sum_i \frac{p(x_i)}{q(x_i)}w_i f(x_i) \tag{6.19}$$

where the samples and weights are computed as described above for Gauss-Hermite quadrature.

The caveat for importance sampling, namely that the proposal density should have support where the target density has support, is somewhat less of a concern in importance quadrature. The reason is that values of x far from the mean are represented in the quadrature rule by points with low weight w_i. The correction factor for importance sampling is $1/q(x)$, but the correction here is $w_i/q(x_i)$, and the weights w_i are increasingly smaller for values of x_i farther from the mean, as shown in Figure 6.2. Figure 6.5 shows the ratio $w_i/q(x_i)$ for a standard normal and 3, 5, 7, 9, and 11 points. Notice that the correction factor remains small, since both w_i and $q(x_i)$ are smaller for values of x_i farther from the mean. The ratio between the largest and smallest value of $w_i/q(x_i)$ is about 1.5, whereas the ratio between the largest and smallest values of $1/q(x)$ shown in Figure 6.4 is on the order of 10^7.

6.3.6 Multidimensional quadrature

The Gaussian quadrature description above assumes that integrals are to be computed over one dimension. For multiple dimensions, we can consider the integral along each dimension and apply a Gaussian quadrature rule to each integral. For two dimensions the integral becomes a double sum,

$$\begin{aligned} \int\int p(x,y)f(x,y)\;dx\;dy &= \sum \int w_i p(y|x_i) f(y;x_i)\;dy \\ &= \sum\sum w_i w_j f(x_i,y_j) \end{aligned} \tag{6.20}$$

and for three dimensions a triple sum

$$\int\int\int p(x,y,z)f(x,y,z)\;dx\;dy\;dz = \sum_i\sum_j\sum_k w_i w_j w_k f(x_i,y_j,z_k) \tag{6.21}$$

Embedding an n point quadrature in d dimensions using this method requires n^d points. If we consider even a minimally solvable SLAM problem with three rover poses and five features, there are $3\times3+5\times2 = 19$ dimensions to the state space. For a three point quadrature rule and 19 dimensions we would require $1,162,261,467$ points, each of which represents an evaluation of the measurement equation. Compared with particle filter methods this represents a few orders of magnitude more samples, and the Unscented filter would require only 39 points. Some benefit may be achieved by considering a 2 point quadrature rule, but even then the number of samples required is $524,288$. The next section describes a few techniques which can be used to reduce the dimensionality of the integrals to be computed so that full multi-dimensional quadrature rules are feasible.

6.4 Dimensionality reduction

We can use the structure of the SLAM problem in order to reduce the dimensionality of the integrals that need to be evaluated in order to compute the posterior statistics. The conditional independences, gauge freedoms, and mixed linear and nonlinear nature of the measurement model can all be exploited in order to reduce the dimensionality.

6.4.1 Conditional independences

The integral in (6.8) is nominally an integral over all of state space, and therefore an integral over potentially many dimensions. We can use sparseness properties of the underlying models to significantly reduce the dimensionality of the integrals to be computed so that they are tractable.

The noise corrupting the measurements are modeled as independent processes. The noise in one bearing measurement is statistically uncorrelated with the noise in another bearing measurement. In other words, the noise covariance **R** is diagonal. Similarly the odometry measurement errors are uncorrelated, but since the odometry measurement $\mathbf{d}_i$ is a vector, the noise covariance **Q** is block diagonal. The noise influencing the bearing and odometry measurements are assumed to be independent as well. Taken as a whole, the noise covariance for all measurements is

$$E\{\nu\nu^T\} = \begin{pmatrix} Q & 0 \\ 0 & R \end{pmatrix} \tag{6.22}$$

which is block diagonal. This also means that the joint measurement likelihood $p(\mathbf{z}|\mathbf{x})$ can be factored into a product of the all of the individual measurement likelihoods

$$p(\mathbf{z}|\mathbf{x}) = \prod_k p(\mathbf{z}_k|\mathbf{x}) \tag{6.23}$$

Furthermore, although the function $\mathbf{h}(\mathbf{x})$ is a multidimensional function of multiple variables, it has significant conditional independences. Figure 1.5 shows a graphical model of the hidden and observable parameters in the bearing only SLAM problem. In the model, an edge between nodes indicates conditional dependence. Lack of an edge indicates that given the parameters which are connected to a node, the value of the node is conditionally independent.

The figure indicates that a bearing measurement b_{ij} is conditionally independent of all other parameters of the state space given the location $\mathbf{s}_j$ of the feature measured and the robot pose $\mathbf{m}_i$ from which it was measured. The odometry measurement $\mathbf{d}_i$ is conditionally independent of all other parameters in state space given the consecutive robot poses $\mathbf{m}_{i-1}$ and $\mathbf{m}_i$ between which the odometry measurement was made. This means that in practice we can partition the distribution $p(\mathbf{z}_k|\mathbf{x})$

6.4.2 Gauge freedoms

We can also exploit the gauge freedoms in the measurement equation to reduce the dimensionality of the integral to be computed. Regardless of the coordinate frame used to compute the expected measurement,

$$\mathbf{h}(\mathbf{m}_i, \mathbf{s}_j) = \arctan(s_{jy} - m_{iy}, s_{jx} - m_{ix}) - m_{i\theta} \tag{6.24}$$

Indeed, the robot may have no idea where it is when it makes the bearing measurements in the first place. Using this fact, we can instead express the measurement equation as

$$\mathbf{h} = \arctan(\Delta y, \Delta x) - m_{i\theta} \tag{6.25}$$

where $\Delta x = s_{jx} - m_{ix}$ and $\Delta y = s_{jy} - m_{iy}$, or

$$\begin{pmatrix} \Delta x \\ \Delta y \\ m_{i\theta} \end{pmatrix} = \begin{pmatrix} -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} m_{ix} \\ m_{iy} \\ m_{i\theta} \\ s_{jx} \\ s_{jy} \end{pmatrix} \tag{6.26}$$

In order to integrate over the new variables, we require the covariance of the state estimate as expressed using the change of variables. If the covariance of $(\mathbf{m}^T \mathbf{s}^T)^T$ is $\mathbf{C}$, then the covariance of $(\Delta x, \Delta y)^T$ is

$$\begin{pmatrix} -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \mathbf{C} \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \tag{6.27}$$

In two dimensional mapping and localization, this change of variables corresponds to a reduction from 5 dimensions down to three. Since the multidimensional quadrature rule requires n^d samples, a 1-D three point quadrature rule can now be embedded using 27 samples instead of 243.

6.4.3 Mixed linear/nonlinear model

The third technique used to reduce the dimensionality of the function to be linearized is to note the mixed linear and nonlinear nature of the measurement function itself. The bearing measurement is of the form

$$\mathbf{h} = \arctan(\Delta y, \Delta x) - m_{i\theta} \tag{6.28}$$

which is nonlinear in Δy and Δx, but linear in $m_{i\theta}$. The derivative

$$\frac{\partial \mathbf{h}}{\partial m_{i\theta}} = -1 \tag{6.29}$$

is valid everywhere. We can use this analytic derivative and then use the statistical linearization approach to linearize $\arctan(\Delta y, \Delta x)$. For 2-D localization and mapping the dimensionality of the function to be linearized has been reduced to 2. A 1-D three point quadrature rule can now be embedded using 9 points rather than 27.

6.5 The algorithm

So far most of this chapter has been devoted to the statistical linearization approach which makes this algorithm unique. The next section presents a step by step explanation of the full algorithm as it applies to the bearing only SLAM problem.

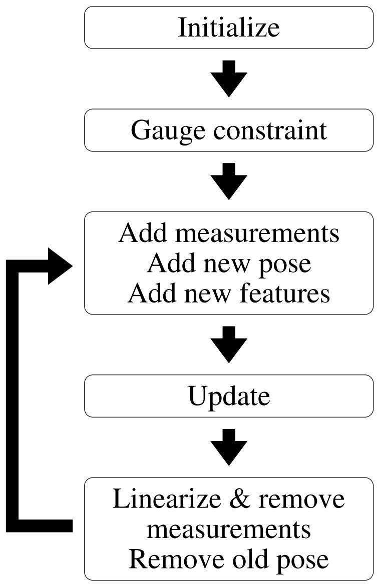


Figure 6.6: Algorithm outline

6.5.1 Initialization

The algorithm is essentially a deterministic nonlinear optimization which recursively estimates the system state. Because the nonlinear optimization is deterministic, global convergence is not guaranteed and a good initial guess is required in order to achieve a reasonable solution. The initialization strategy consists of three steps. The first step is to integrate the odometry to get an initial trajectory. The second step is to use the initial trajectory and the bearing measurements in a closed-form linear estimation step to initialize the map. The third step is to optimize the trajectory and map with respect to all of the measurements in a full nonlinear optimization.

In the bearing only SLAM problem, robot motion information is available in the form of odometry. Motion estimation from odometry is insufficient for long durations but over short durations odometry can provide a reasonable estimate of the robot trajectory. The odometry is of the form $d_i = (\Delta x_i, \Delta y_i, \Delta\theta_i)$, specified in the rover coordinate frame at the start of the motion. Dead reckoning, or path integration, is achieved by starting with $\mathbf{m}_0 = (0,0,0)^T$ and then iteratively computing

$$\mathbf{m}_i = \begin{pmatrix} m_{i-1,x} + \cos(m_{i-1,\theta})\Delta x_i - \sin(m_{i-1,\theta})\Delta y_i \\ m_{i-1,y} + \cos(m_{i-1,\theta})\Delta y_i + \sin(m_{i-1,\theta})\Delta x_i \\ m_{i-1,\theta} + \Delta\theta_i \end{pmatrix} \tag{6.30}$$

With an initial estimate for robot trajectory plus a set of bearings, we can set up a linear estimation problem for the position of a feature. The position of each feature can be independent determined knowing the trajectory and the bearing toward that feature from at least two different points along the trajectory. Setting up the problem as a linear estimation problem guarantees that there is a unique solution, which means that if an initial guess for the trajectory can be made with odometry, then the best initial map can be computed directly and without the annoyance of local minima.

The relationship between a feature and the robot can be expressed in terms of the robot pose, feature position, and an unknown depth λ_{ij} in the following two equations:

$$\begin{aligned} s_{jx} &= m_{ix} + \lambda_{ij}\cos(m_{i\theta} + b_{ij}) \\ s_{jy} &= m_{iy} + \lambda_{ij}\sin(m_{i\theta} + b_{ij}) \end{aligned} \tag{6.31}$$

where m_{ix}, m_{iy}, and $m_{i\theta}$ have already been estimated using odometry and are therefore known quantities. We need only estimate s_{jx}, s_{jy}, which are the parameters of interest, and λ_{ij}, a nuisance parameter. Equations (6.31) represent two constraints and three unknowns. Two additional equations and one additional unknown are introduced for each additional pose $\mathbf{m}_{i+1}$. In general, bearing measurements for feature j from M different poses can be used to generate $M + 2$ unknowns and $2M$ equations, so that with at least two different poses there exists a unique solution.

Note that since the rover orientation $m_{i\theta}$ is held fixed and the bearing b_{ij} is a direct measurement, equations (6.31) are linear in the unknowns. Let $\phi_{ij} = m_{i\theta} + b_{ij}$. Then by rearranging (6.31) we get the following system of equations

$$\begin{pmatrix} 1 & 0 & -\cos(\phi_{1j}) & 0 & \cdots & 0 \\ 0 & 1 & -\sin(\phi_{1j}) & 0 & \cdots & 0 \\ 1 & 0 & 0 & -\cos(\phi_{2j}) & \cdots & 0 \\ 0 & 1 & 0 & -\sin(\phi_{2j}) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & 0 & 0 & 0 & \cdots & -\cos(\phi_{Mj}) \\ 0 & 1 & 0 & 0 & \cdots & -\sin(\phi_{Mj}) \end{pmatrix} \begin{pmatrix} s_{jx} \\ s_{jy} \\ \lambda_{1j} \\ \lambda_{2j} \\ \vdots \\ \lambda_{Mj} \end{pmatrix} = \begin{pmatrix} m_{1x} \\ m_{1y} \\ m_{2x} \\ m_{2y} \\ \vdots \\ m_{Mx} \\ m_{My} \end{pmatrix} \tag{6.32}$$

This system of equations can be solved directly in a least squares sense, yielding an estimate of the feature position $\mathbf{s}_j$ along with the nuisance parameters λ_{ij}. This linear system is set up and solved independently for each feature j.

The solution to (6.32) may actually correspond to a feature location which is on the wrong side of the rover. Such a solution is easy to detect, since the depth parameters λ_{ij} will be negative. This can easily happen if the measurements toward the feature diverge as in Figure 6.7. If this is the case, the feature can be removed from the filter, or simply held out of the estimation until further measurements are available which provide a more reasonable estimate. In many cases, the initial guess obtained by solving (6.32) is very far off but enough features are estimated properly that the nonlinear optimization will still succeed. However, a conservative approach would remove the feature and the corresponding measurements.

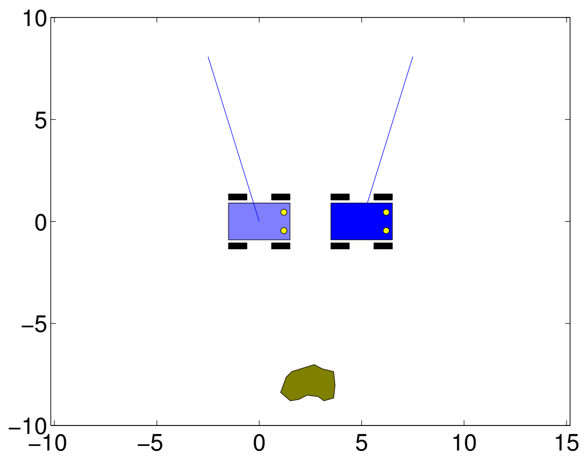


Figure 6.7: Initialization with divergent bearings can lead to poor feature position estimates.

The next step is to use the initial trajectory obtained via the path integration in equation (6.30) and the initial map obtained by solving (6.32) as an initial guess for Gauss-Newton optimization. Together the trajectory and map constitute the state vector

$$\mathbf{x} = \begin{pmatrix} \mathbf{m} \\ \mathbf{s} \end{pmatrix} \tag{6.33}$$

and the odometry and bearings constitute the measurement vector

$$\mathbf{z} = \begin{pmatrix} \mathbf{d} \\ \mathbf{b} \end{pmatrix} \tag{6.34}$$

and the state is optimized with respect to the measurements by minimizing the squared error (or robust norm) with respect to the measurements predicted by the measurement model

$$e = (\mathbf{z} - \mathbf{h}(\mathbf{x}))^T \mathbf{R}^{-1} (\mathbf{z} - \mathbf{h}(\mathbf{x})) \tag{6.35}$$

This cost function is minimized using Gauss-Newton, which requires the computation of the Jacobian and Hessian of (6.35)

$$\begin{aligned} \epsilon &= \nabla e = -\mathbf{H}^T \mathbf{R}^{-1} (\mathbf{z} - \mathbf{h}(\mathbf{x})) \\ \mathbf{A} &= \nabla^2 e = \mathbf{H}^T \mathbf{R}^{-1} \mathbf{H} \end{aligned} \tag{6.36}$$

and then solving

$$\begin{aligned} \mathbf{A}\delta &= \epsilon \\ \mathbf{x} &\leftarrow \mathbf{x} - \delta \end{aligned} \tag{6.37}$$

Once the optimization converges to some value of the state vector, the algorithm is initialized. The Hessian computed using (6.36) at the minimum is the initial inverse covariance for the filter.

6.5.2 Gauge constraint

As discussed in Section 1.3.3, the bearing only SLAM problem contains a gauge freedom in the coordinate frame. Both the location of the origin (translation) and the direction of the x and y axes (rotation) are undetermined by the odometry and bearing measurements alone. Unlike the SFM problem, the scale factor is disambiguated by odometry, since the number of wheel rotations measured in each interval provide redundant information about the total distance traveled.

In order to constrain the rotational and translational gauge freedoms, we simply constrain the solution to use an initial robot pose of $(0, 0, 0)^T$. This constraint means that the uncertainty along the dimensions of the state space corresponding to the initial rover pose is zero.

Applying the constraint and removing the first robot pose (with absolute certainty) from the filter results in an inverse covariance of A where

$$A_0 = \begin{matrix} C & B \\ B^T & A \end{matrix} \tag{6.38}$$

is the inverse covariance from the optimization step.

6.5.3 Adding states and measurements

New measurements and new states must be added to the filter. Odometry measurements are always added to the measurement vector. The odometry measurement is also used to predict the next robot pose. One step of the path integration in equation (6.30) is done to get an initial estimate of the current robot pose $\mathbf{m}_i$. This new robot pose is inserted into the state vector so that

$$\mathbf{x} = \begin{pmatrix} \mathbf{m}_{i-L+1} \\ \mathbf{m}_{i-L+2} \\ \vdots \\ \mathbf{m}_{i-1} \\ \mathbf{s} \end{pmatrix}, \qquad \mathbf{x} = \begin{pmatrix} \mathbf{m}_{i-L+1} \\ \mathbf{m}_{i-L+2} \\ \vdots \\ \mathbf{m}_{i-1} \\ \mathbf{m}_i \\ \mathbf{s} \end{pmatrix} \tag{6.39}$$

Bearing measurements corresponding to features already in the filter are added, but bearings corresponding to new features are first put into a tentative target list, as in the approach described by Dissanayake et al.[30] Once the target has been observed a few times, the current trajectory estimate for the robot is combined with the measurements of the tentative feature using the linear least squares approach described in Section 6.5.1. If this initialization procedure finds a reasonable position for the feature, i.e. if the recovered depth parameters λ_{ij} are all positive and the squared error residual is low, then the feature position

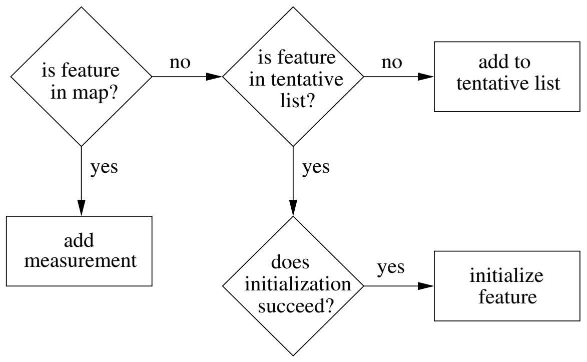


Figure 6.8: Steps to manage target list and initialize features

$\mathbf{s}_j$ is inserted into the state vector so that

$$\mathbf{x} = \begin{pmatrix} \mathbf{m} \\ \mathbf{s}_0 \\ \mathbf{s}_1 \\ \vdots \\ \mathbf{s}_{j-1} \end{pmatrix}, \qquad \mathbf{x} \leftarrow \begin{pmatrix} \mathbf{m} \\ \mathbf{s}_0 \\ \mathbf{s}_1 \\ \vdots \\ \mathbf{s}_{j-1} \\ \mathbf{s}_j \end{pmatrix} \tag{6.40}$$

and the bearing measurements are inserted into the measurement vector.

Figure 6.9 shows an example. A new feature is being initialized after enough measurements have been made to confirm the feature. Eight features already exist in the map, shown with uncertainty ellipses around the estimated locations. The bearings toward the ninth feature from the current estimated trajectory are shown as blue lines. The closed form solution is obtained for the position of the feature and it is inserted into the map.

Note that the initialization of new features used here does not find the optimal solution. The initialization procedure estimates a feature position which is good enough to seed nonlinear optimization. Information from the feature measurements will help improve the pose estimates from which the feature was measured. Rather than linearize the measurements after computing this rough initial position estimate as is done in some approaches[75], we leave the measurements in nonlinear form and proceed to update the state using all of the available information.

6.5.4 State update

After the new measurements have been added to the measurement vector, the new pose has been added to the trajectory and any confirmed features have been added to the map, the algorithm performs an optimization step. The optimization step uses Gauss-Newton to update the state with respect to the

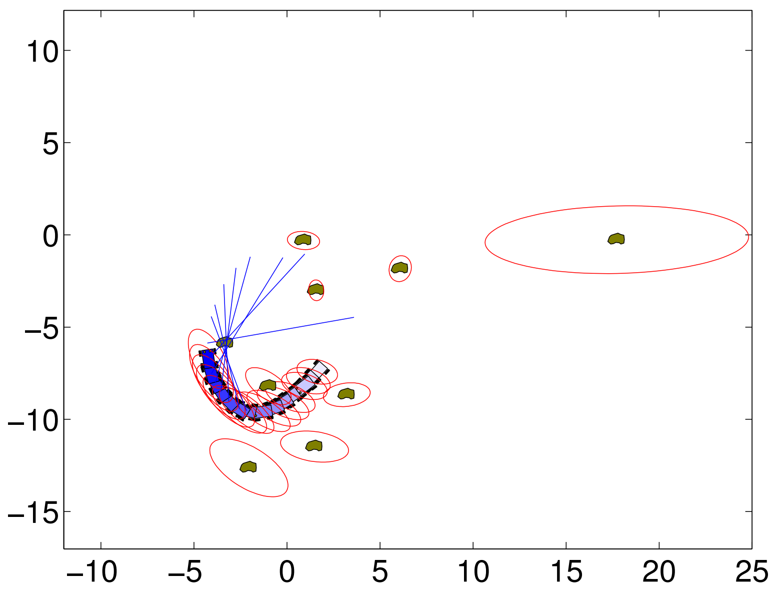


Figure 6.9: New feature being initialized using closed form mapping technique

prior, modeled with $\mathbf{x}_0$ and $\mathbf{A}_0$, and the measurement likelihood. The cost function to be minimized is

$$e = (\mathbf{x} - \mathbf{x}_0)^T \mathbf{A}_0 (\mathbf{x} - \mathbf{x}_0) + (\mathbf{z} - \mathbf{h}(\mathbf{x}))^T \mathbf{R}^{-1} (\mathbf{z} - \mathbf{h}(\mathbf{x})) \tag{6.41}$$

The minimization proceeds as before, by computing the derivatives

$$\begin{aligned} \epsilon &= \nabla e = \mathbf{A}_0(\mathbf{x} - \mathbf{x}_0) - \mathbf{H}^T \mathbf{R}^{-1} (\mathbf{z} - \mathbf{h}(\mathbf{x})) \\ \mathbf{A} &= \nabla^2 e = \mathbf{A}_0 + \mathbf{H}^T \mathbf{R}^{-1} \mathbf{H} \end{aligned} \tag{6.42}$$

and then solving

$$\begin{aligned} \mathbf{A}\delta &= \epsilon \\ \mathbf{x} &\leftarrow \mathbf{x} - \delta \end{aligned} \tag{6.43}$$

Once the optimization converges to some value of the state vector, the update is complete. This step is very similar to the update in the iterated extended Kalman filter (IEKF), except that a history of measurements is used for the update rather than only the most recent measurement.

6.5.5 Removing old measurements

Removing old measurements is done by computing the contributions to the derivative ϵ and Hessian $\mathbf{A}$ of the cost function in (6.42). To remove measurement j, we apply the statistical linearization method covered in Section 6.2. As explained in that section, the coefficients **h**1 and h_0 are sought which minimize

$$e = E_{p(\mathbf{x}|\mathbf{z})}\{(z_j - h_j(\mathbf{x}) - \mathbf{h}_1^T\mathbf{x} + h_0)R_j^{-1}(z_j - h_j(\mathbf{x}) - \mathbf{h}_1^T\mathbf{x} + h_0)\} \tag{6.44}$$

by solving the linear system

$$\begin{pmatrix} E_{p(\mathbf{x}|\mathbf{z})}\{\mathbf{x}\mathbf{x}^T\} & -E_{p(\mathbf{x}|\mathbf{z})}\{x\} \\ -E_{p(\mathbf{x}|\mathbf{z})}\{\mathbf{x}^T\} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{h}_1 \\ h_0 \end{pmatrix} = \begin{pmatrix} E_{p(\mathbf{x}|\mathbf{z})}\{\mathbf{x}(z_j - h_j(\mathbf{x}))\} \\ -E_{p(\mathbf{x}|\mathbf{z})}\{(z_j - h_j(\mathbf{x}))\} \end{pmatrix} \tag{6.45}$$

The individual terms in (6.45) are integrated using the importance quadrature and dimensionality reduction techniques discussed in Sections 6.3 and 6.4, and then (6.45) is solved for $\mathbf{h}_1$ and h_0.

Once the linearization is computed, the approximation can be made. The log prior from the last time step is combined with the linearized measurement model

$$e = (\mathbf{x} - \mathbf{x}_0)^T\mathbf{A}_0(\mathbf{x} - \mathbf{x}_0) + (\mathbf{h}_1^T\mathbf{x} - h_0)^T R_j^{-1}(\mathbf{h}_1^T\mathbf{x} - h_0) \tag{6.46}$$

where h_0 and $\mathbf{h}_1$ now relate the full state $\mathbf{x}$ to the linearized measurement, which means that the nonzero components are in the columns corresponding to the state variables which influence measurement z_j and they contain zeros for all other entries. The two terms in (6.46) are both quadratic in $\mathbf{x}$. The mean and inverse covariance of the new posterior can be computed using

$$\begin{aligned} \frac{\partial e}{\partial \mathbf{x}} &= \mathbf{A}_0(\mathbf{x} - \mathbf{x}_0) - \mathbf{h}_1 R_j^{-1}(\mathbf{h}_1^T\mathbf{x} - h_0) \\ \frac{\partial^2 e}{\partial \mathbf{x}^2} &= \mathbf{A}_0 + \mathbf{h}_1 R^{-1}\mathbf{h}_1^T \end{aligned} \tag{6.47}$$

The new mean of (6.46) is computed as

$$\mathbf{x}_0 \leftarrow \mathbf{x}_0 - (\mathbf{A}_0 + \mathbf{h}_1 R_k^{-1}\mathbf{h}_1^T)^{-1}\mathbf{h}_1 R^{-1}(\mathbf{h}_1^T\mathbf{x}_0 - h_0) \tag{6.48}$$

and the new inverse covariance is computed as

$$\mathbf{A}_0 \leftarrow \mathbf{A}_0 + \mathbf{h}_1 R^{-1}\mathbf{h}_1^T \tag{6.49}$$

It can be seen immediately that one bearing measurement provides the addition of a rank one matrix to the inverse covariance $\mathbf{A}_0$, since the update in (6.49) adds a single vector outer product.

Once the measurements have been linearized and the likelihood terms incorporated into the prior, the pose corresponding to the bearing and odometry measurements that were removed is also removed. The state vector is simply transformed by removing the state variables corresponding to the old pose

$$\mathbf{x} = \begin{pmatrix} \mathbf{m}_{i-L+1} \\ \mathbf{m}_{i-L+2} \\ \vdots \\ \mathbf{m}_i \\ \mathbf{s} \end{pmatrix}, \qquad \mathbf{x} = \begin{pmatrix} \mathbf{m}_{i-L+2} \\ \vdots \\ \mathbf{m}_i \\ \mathbf{s} \end{pmatrix} \tag{6.50}$$

and the inverse covariance is updated by marginalizing the covariance. If the inverse covariance is partitioned into blocks corresponding to the states to be removed and the states to be left in the filter,

$$\mathbf{A}_0 = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix} \tag{6.51}$$

then from the matrix inversion lemma for block matrices, the marginal inverse covariance is computed by

$$A_0 \leftarrow A_{22} - A_{21}A_{11}^{-1}A_{12} \tag{6.52}$$

A similar procedure may be followed for removing features from the map. This might be done for reasons such as limiting the complexity of the algorithm by concentrating only on local features[28]. However, in the current implementation of the work in this thesis, features are not removed.

The next step of the algorithm is to return to the measurement and state insertion step described in Section 6.5.3.

6.6 Robust nonlinear smoothing

The Variable State Dimension Filter was designed as a best approximation to full nonlinear least squares. In order to include robustness in the algorithm, this idea can be extended to a nonlinear M-estimator using the Iteratively Reweighted Least Squares (IRLS) approach described in 3.3. The smoothing section of the VSDF can include iterative reweighting of each measurement based on the innovation, as in IRLS. When a measurement is linearized, the weight for that measurement is fixed. The linear and quadratic contributions of the measurement to the Jacobian and Hessian of estimation term are computed one last time, then held fixed and incorporated into the posterior. Measurements which cannot be consistently incorporated into the posterior will have small weights. This downweighting of inconsistent measurements provides some robustness to outlying measurements. The robust recursive smoother is described by the following cost function and its derivatives

$$\begin{aligned} e &= (\mathbf{x}-\mathbf{x}_0)^T\mathbf{A_0}(\mathbf{x}-\mathbf{x}_0) + \sum w(\epsilon_k)\epsilon_k^T\mathbf{R_k}^{-1}\epsilon_k \\ \mathbf{a} &= \mathbf{a_0} + \mathbf{A_0}(\mathbf{x}-\mathbf{x}_0) + \sum w(\epsilon_k)\mathbf{H_k}^T\mathbf{R_k}^{-1}\epsilon_k \\ \mathbf{A} &= \mathbf{A_0} + \sum w(\epsilon_k)\mathbf{H_k}^T\mathbf{R_k}^{-1}\mathbf{H_k} \end{aligned} \tag{6.53}$$

where $\epsilon_k = \mathbf{z_k} - \mathbf{h_k}(\mathbf{x})$ is the residual of measurement k and the weight function $w(\epsilon_k)$ is one of the weighting functions from Table 3.2.

6.7 Square root filtering

Finally, a reduction in computational complexity can be realized by working with the Cholesky factorization of the Hessian matrix rather than the Hessian

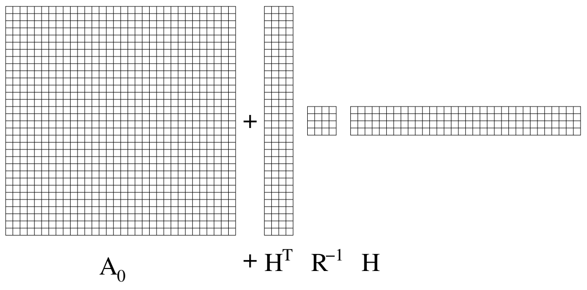


Figure 6.10: Pictorial representation of the Hessian update

itself. Cholesky factorization is a common means of solving a system of linear equations $Ax = b$ when the coefficient matrix A is symmetric and positive definite. The Cholesky factorization $\mathbf{SS}^T = \mathbf{A}$ is first computed, then the two triangular systems

$$\begin{aligned} \mathbf{Sy} &= \mathbf{b} \\ \mathbf{S}^T\mathbf{x} &= \mathbf{y} \end{aligned} \tag{6.54}$$

are solved. The Cholesky decomposition of a dense $N \times N$ matrix can be computed in $O(N^3)$. The solution to the two triangular linear systems is $O(N^2)$.

If rather than storing and manipulating the full Hessian matrix $\mathbf{A}$ we can store and manipulate its Cholesky factor S, then the factorization step can be avoided and the solution to (5.37) can be computed with $O(N^2)$ backsubstitutions alone. The only remaining problem is to determine how to propagate the Cholesky factor from step to step in the filter.

There exist algorithms for performing the *update* and *downdate* of a Cholesky factor, where update is defined as the addition of a symmetric outer product $\mathbf{A}' = \mathbf{A} + v^T \sigma v$ and downdate is the subtraction of a symmetric outer product $\mathbf{A}' = \mathbf{A} - v^T \sigma v$. For a rank-1 update or downdate these algorithms require $O(N^2)$ computation, so a rank-k update can be done in $O(kN^2)$. If we already have the Cholesky factor for the prior Hessian $\mathbf{A_0}$, then the step which combines the prior and the likelihood is

$$A = A_0 + \mathbf{H}^T R^{-1} \mathbf{H} + F^T Q^{-1} F \tag{6.55}$$

which can be computed as a Cholesky update, and the marginalization of state dimensions to be removed from the filter is

$$A_0' = A_0(2:n, 2:n) - A_0(2:n, 1) A_0(1,1)^{-1} A_0(1, 2:n) \tag{6.56}$$

which can be computed as a Cholesky downdate. A pictorial representation of the matrix operation in a Cholesky update is in Figure 6.10. Typically $\mathbf{H}$ is rectangular and there are far more columns (state dimensions) than rows (measurements in the nonlinear portion of the filter) so that the Cholesky updating procedure is more efficient than the straightforward full Hessian implementation.

Since we can perform Cholesky updates and downdates for adding and removing measurements and states in the filter, and use the Cholesky factors in the optimization step to solve the normal equations, we can do away with the full covariance matrix $\mathbf{A}$ and only maintain and use its Cholesky decomposition $\mathbf{S}$. This technique has been used to modify the EKF for parameter estimation problems[54].

The insertion and removal of measurements and states in the filter proceeds as before except that Cholesky updates and downdates replace the operations on $\mathbf{A_0}$.

6.8 Conclusions

This chapter presents a novel method for bearing only SLAM. The technique derived relies on the VSDF[57], a nonlinear smoothing technique which was originally derived for Structure from Motion. The smoothing algorithm can be improved by considering the computation of posterior statistics. The statistical linearization criterion can be applied to compute a linearization, and the criterion requires the computation of an integral which does not have an analytical solution.

By using the Gauss-Hermite quadrature rule from numerical integration, and modifying the rule to use a corrective weighting similar to importance sampling from Monte Carlo, the integral can be converted to a sum which approximates the integral. Because the full quadrature rule requires a sample size which is exponential in the number of dimensions, it is necessary to exploit the nature of the problem to reduce the dimensionality. The conditional independence of measurements, gauge freedoms, and mixed linear and nonlinear nature of the measurement models can be used to reduce the dimensionality from hundreds or thousands down to two or three.

Finally, a robust version of the recursive smoothing filter is achievable by extending the smoothing algorithm to be a recursive version of robust M-estimators rather than a recursive version of total least squares.

In the next chapter, results from applying the algorithm to synthetic and real data will be shown.

Chapter 7

Experiments

This chapter presents experimental support for the claims put forth by this dissertation. In some cases, synthetic examples are preferred. The ability to precisely control the experimental setup, to know the ground truth for all of the state variables, to generate noise from a known distribution, and to repeat experiments with similar conditions but different corrupting noise makes it possible to quantify the performance of the algorithm. However, “simulations are doomed to succeed,” so experiments with real data are included to verify that the techniques can successfully move out of the lab and into the real world.

7.1 Experiments with synthetic data

7.1.1 Comparison to Kalman filters

This first set of experiments investigate the difference in performance between a standard EKF and the new method. There are three results presented which characterize this difference. These results are from applying the two methods to the simulated example in Figure 7.1, and computing the average map error (sum of distances between the true and estimated map) over time for 100 Monte Carlo runs. Each Monte Carlo run is carried out by first generating synthetic data for the ground truth case, then running each of the estimators on the same synthetic data and recording the resulting map errors.

The first experiment compares the result of localizing and mapping using both methods with all landmarks visible from the all positions along the traverse. This decouples issues of initialization of landmark positions from the convergence properties of the two filters. The experiment is repeated with two different noise levels. The results in Figure 7.2 shows that the new method converges more rapidly and to a better final solution than an EKF. This is probably partly because the EKF must compute posterior statistics with only the first measurement and some guess as to the range to the target. In this experiment the initial range used was the average range to the landmarks. In

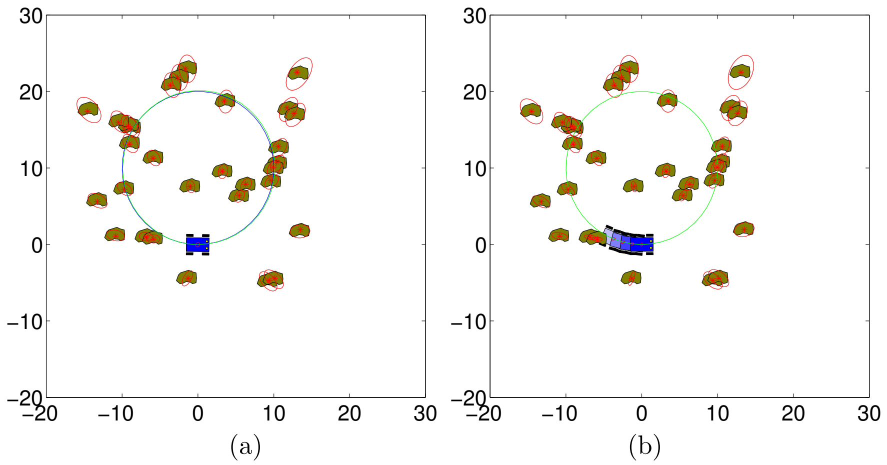


Figure 7.1: Map and trajectory used to generate synthetic data. (a) shows a typical result from the EKF and (b) shows a result from the new method.

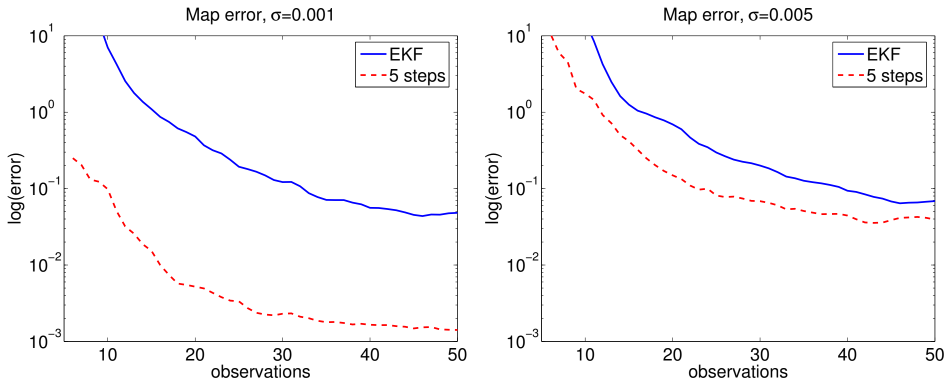


Figure 7.2: Map errors over time for the EKF and the new method. (a) and (b) show the sum of distances between the ground truth and estimated map with different levels of noise.

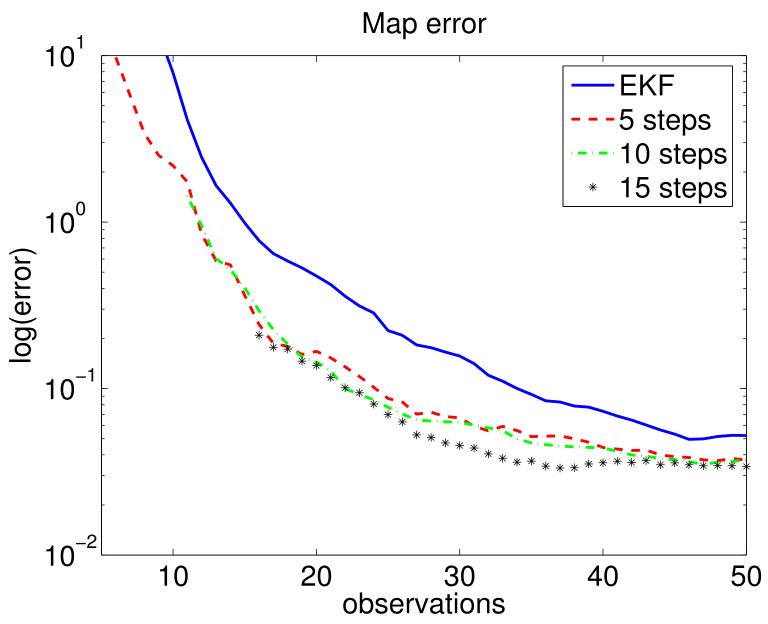


Figure 7.3: Map errors over time for the EKF and the new method with different smoothing windows.

real applications an initial guess would be necessarily less favorable since the average distance to new targets could not be known.

The second experiment compares the results using different numbers of steps in smoothing. The data is again from a simulation where all landmarks are visible from all positions to demonstrate that the improvement comes in part simply from the computation of statistics using more data. The EKF uses the fewest fully nonlinear models per update, and is compared to smoothing windows of 5, 10, and 15 steps. It is expected that a larger window of observations in the smoothing filter will yield improved results, since more data is used to compute the state estimate. Figure 7.3 demonstrates this.

The third experiment compares results from an EKF and smoothing filter where landmarks are not all seen from all locations. This introduces the problem of initializing landmarks as they are first observed and computing the map and trajectory using a smaller number of local observations. Figure 7.4 shows that the EKF has much more trouble with this scenario. New landmarks are being observed frequently, and inserted into the map in the direction of the first bearing measurement at a fixed distance from the robot location. The distance chosen was the average distance to new targets, which in real applications would not necessarily be known at all. Using other choices for the initial distance yielded worse results. The smoothing filter however performs nearly as well as the scenarios above where all features are visible from all locations.

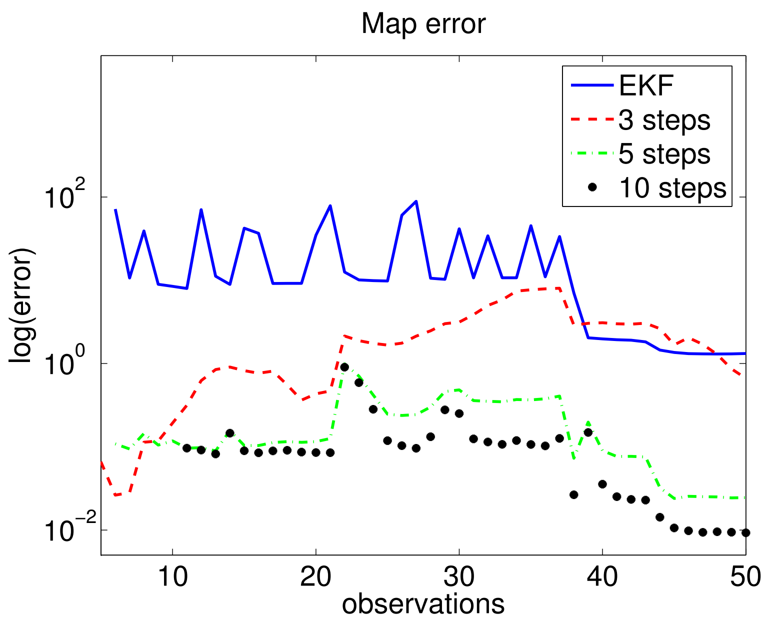


Figure 7.4: Map errors over time for the EKF and the new method, where new landmarks become visible and must be initialized.

7.1.2 Comparison of linearization methods

The following example compares the mapping errors of the statistical linearization method for computing posterior statistics with the typical Jacobian based linearization. Figure 7.5 shows a robot trajectory and map, where only every fifth robot pose is plotted for clarity. This example has 89 robot poses and four landmarks. This case is used to illustrate the difference between linearization methods since the small number of landmarks makes the problem less well posed. With less data it is more important to approximate the data accurately.

Figure 7.6 shows the error in the reconstructed map over time for 100 Monte Carlo runs. The two methods are comparable for about the first thirty time steps, but at $t = 40$ the reconstruction error for the Jacobian based method begins to rise, and by $t = 89$ the reconstruction error is on average significantly higher than for the quadrature based method. This is not surprising. It is known that the EKF is biased, and the Jacobian based method for computing posterior statistics is similar to an EKF. However, the statistical linearization produces posterior statistics which capture the shape of the posterior in a larger neighborhood, so that evidence which is gathered later does not have as drastic an effect on the estimate. The result is that the variance of the mapping error is much smaller.

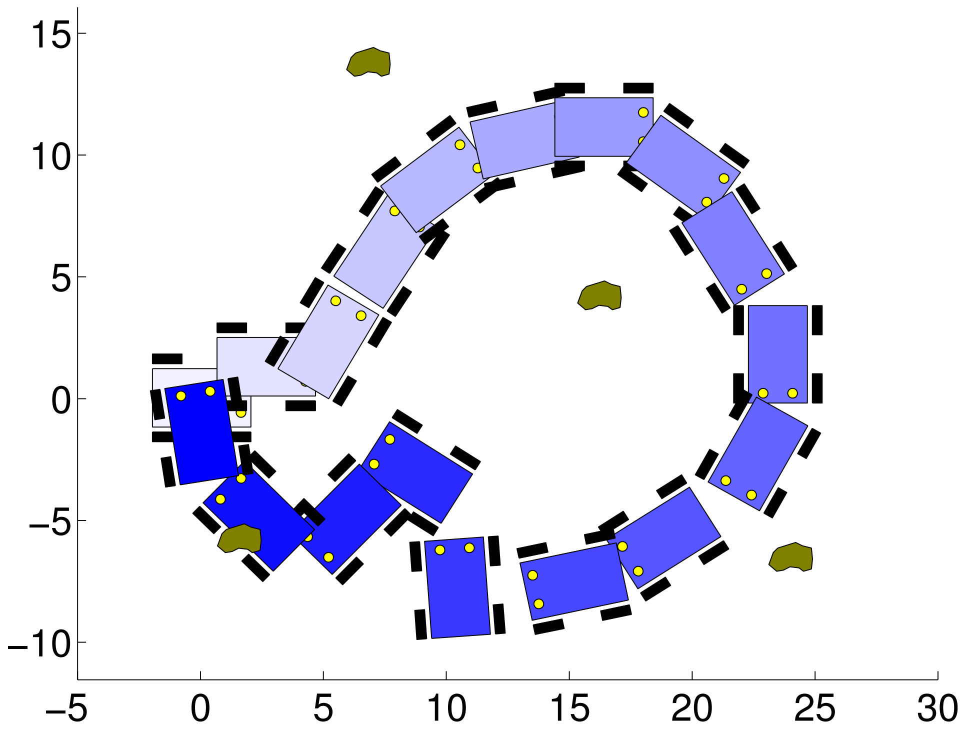


Figure 7.5: Map and trajectory used to generate synthetic data.

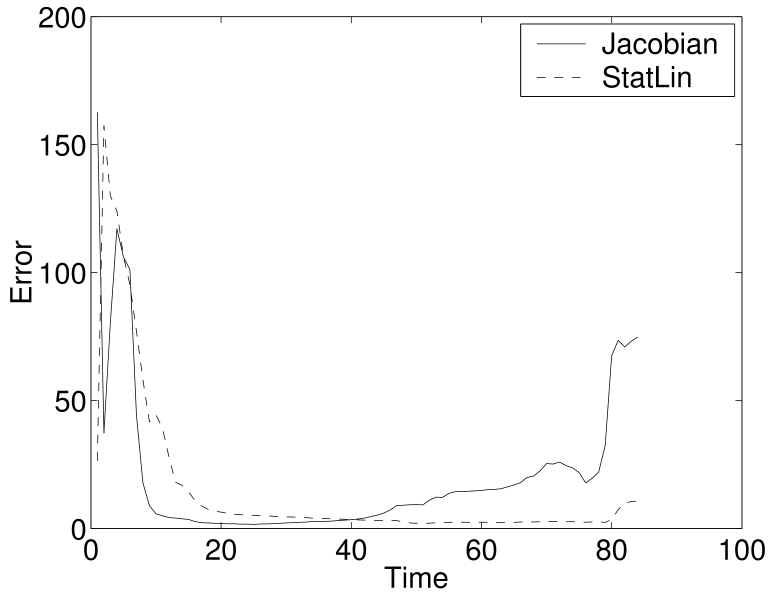


Figure 7.6: Average reconstruction error for Jacobian and statistical linearization methods.

7.1.3 Consistency of the algorithm

The next experiment is to compare the consistency of the filter using the Jacobian linearization and the statistical linearization. Figure 7.7 shows the ground truth model and trajectory. For each linearization method, 50 Monte Carlo runs were completed. The smoothing algorithm used a lag of 5 timesteps in order to make the processing more difficult. The data was generated by evaluating the measurement models for odometry and bearing measurements and corrupting the measurements by adding independent Gaussian noise. Both filters were run on the same data samples so that the comparison is not influenced by processing particularly difficult data with only one of the methods. Figures 7.8 and 7.9 show the result of the 50 trials with each filter using a time lag of 5. The result computed using full batch estimation on uncorrupted data is plotted along with uncertainty ellipses in order to get a sense of the posterior covariance evaluated near the maximum likelihood solution. The Jacobian based method clearly exhibits divergence, since many of the trajectory and map estimates shown in Figure 7.8 are well outside the covariance ellipses. The statistical linearization, on the other hand, exhibits consistency since the map and trajectory estimates in Figure 7.9 mostly lie within the uncertainty ellipse. Since the ellipse corresponds to a 99% confidence interval, it is not unreasonable for some of the points to lie outside the ellipses. For 100 runs, we can expect one result to lie outside the interval, and since the ellipses correspond to marginal covariances, it is not surprising that one run of the algorithm can produce estimates for all

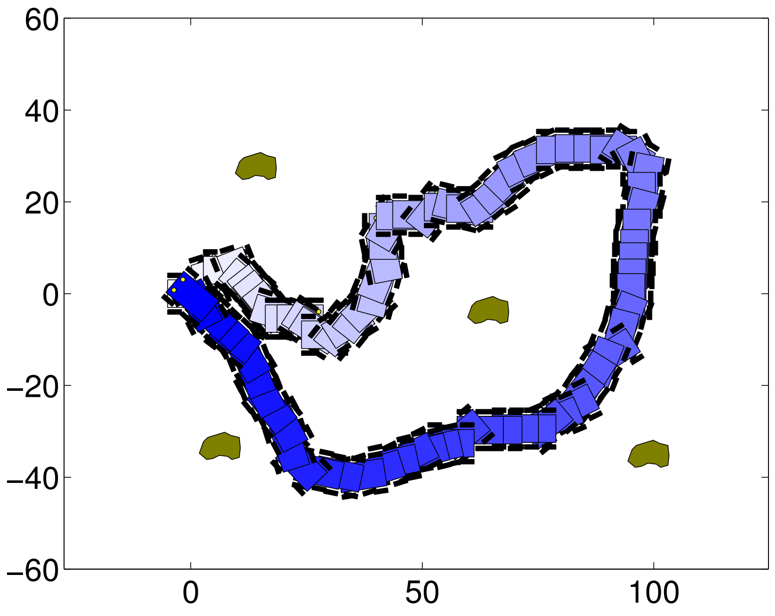


Figure 7.7: Ground truth for consistency experiment.

of the map features and robot poses which lie outside the intervals since they are in fact correlated.

It should be noted that both of the methods will improve by using a longer lag time. Figure 7.10 shows the result of running 50 Monte Carlo runs and Jacobian linearization with a time lag of 10 rather than 5. The results are improved but still show some divergence. Figure 7.11 shows 50 Monte Carlo runs with the same data and a time lag of 10 with the statistical linearization. The result has lower variance than the result with statistical linearization and a time lag of 5 and again shows good consistency.

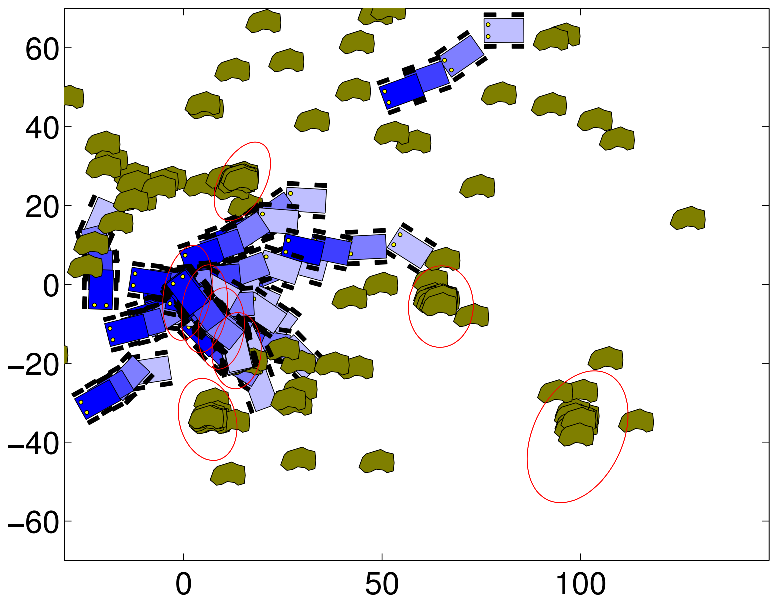


Figure 7.8: Monte Carlo runs with a lag of 5 and Jacobian linearization.

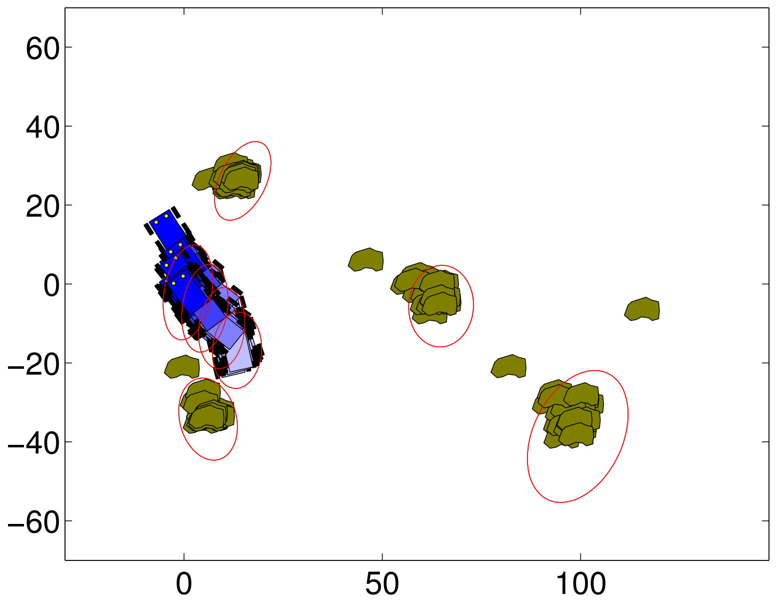


Figure 7.9: Monte Carlo runs with a lag of 5 and statistical linearization.

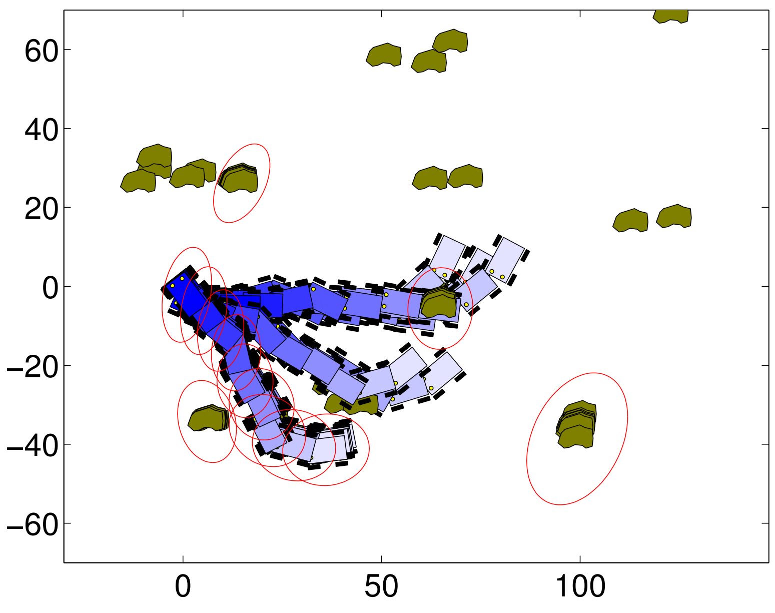


Figure 7.10: Monte Carlo runs with a lag of 10 and Jacobian linearization.

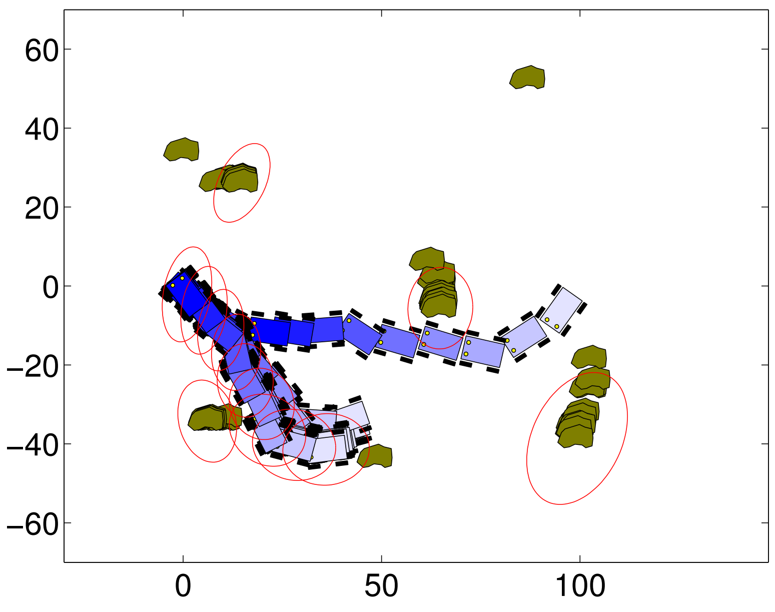


Figure 7.11: Monte Carlo runs with a lag of 10 and statistical linearization.

7.2 Experiments with real data

7.2.1 Grommit in the SPA

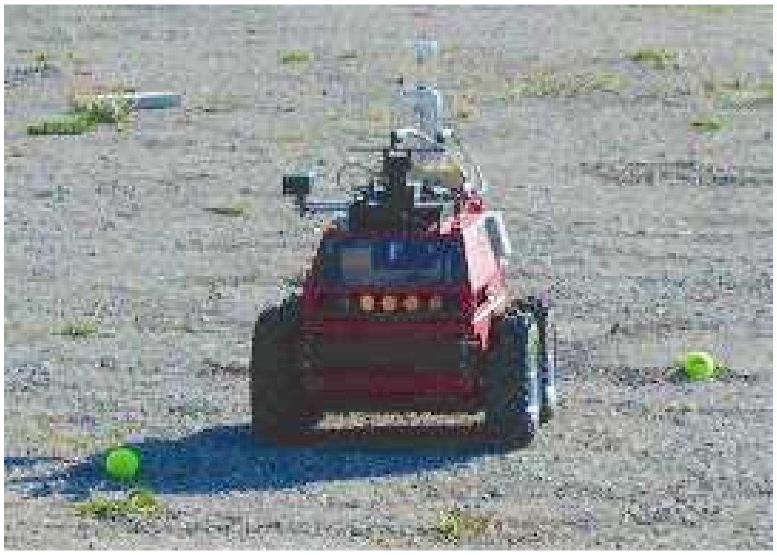


Figure 7.12: NASA Ames ATRV Junior, “grommit”, used to collect data for experiments

The next experiment uses data from a real robot. NASA Ames Research Center has an ActivMedia ATRV Junior called “grommit”. Grommit was used to collect data for navigation experiments in the Science Planetary Analog (SPA) site at Ames. A photograph of grommit in the SPA is in Figure 7.12.

An omnidirectional camera was attached to grommit and the robot was driven in a gravel lot, acquiring omnidirectional images once approximately every five seconds. A seven minute traverse yielded 91 images as the robot drove in a loop, stopping close to its starting point. The rover logged wheel odometry once every second, and images once every five seconds. Figure 7.13 shows the result of path integration using only odometry.

Figure 7.14 shows every 10^{th} frame of the video sequence captured with the omnidirectional camera. The red dots in the images mark the image plane locations of tracked features.

The overall trajectory traveled consisted of about 50 meters driving distance. Ground truth position is not available, but at the end of the traverse, the rover was approximately back at the starting point with approximately the same heading that it started with. The dead reckoning estimate results in a final position error of about 4.03 meters, or 10% of distance travelled, and a final heading error of about 55^o. The batch estimate recovers a final position within 40cm of the initial starting point, and a heading within 4^o. Figure 7.18 shows the

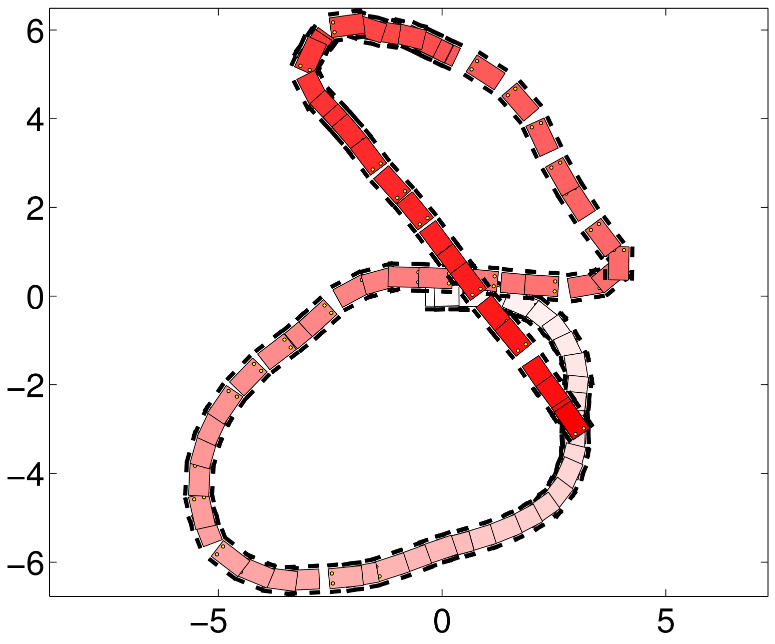


Figure 7.13: Dead reckoning for grommit.

trajectory and map estimated using full batch estimation. Figure 7.16 shows a comparison of the trajectory computed using dead reckoning and the trajectory computed using batch estimation.

Figure 7.17 shows the final trajectory segment and map estimated by the algorithm presented in Chapter 6. The trajectory segment corresponds to the rover poses still in the nonlinear section of the smoothing algorithm, i.e. those which have not been linearized and removed. This includes the final pose of the rover, which was estimated to be $(m_x = 0.405, m_y = -0.625, m_\theta = 5.3^o)$ with respect to the first pose. Compared to a final pose estimate of $(m_x = 0.273, m_y = -0.305, m_\theta = 4.1^o)$ from the full batch estimate, the result from the nonlinear smoothing is relatively close and represents a significant improvement over odometry alone.

Figure 7.19 shows the result from the nonlinear smoothing with the result from full batch estimation. The red corresponds to the batch estimate and the blue corresponds to the nonlinear smoothing algorithm. The discrepancy between the output of the two estimators is well within the uncertainty ellipses, as expected. With only one example, this is not a sufficient test for consistency. Measurements are only available from one run, however, so ensemble averages would need to be computed by generating multiple sets of synthetic data from the real data that was collected.

One interesting result from this experiment can be seen by looking at feature 11 and feature 16. These are actually the same feature but are labelled

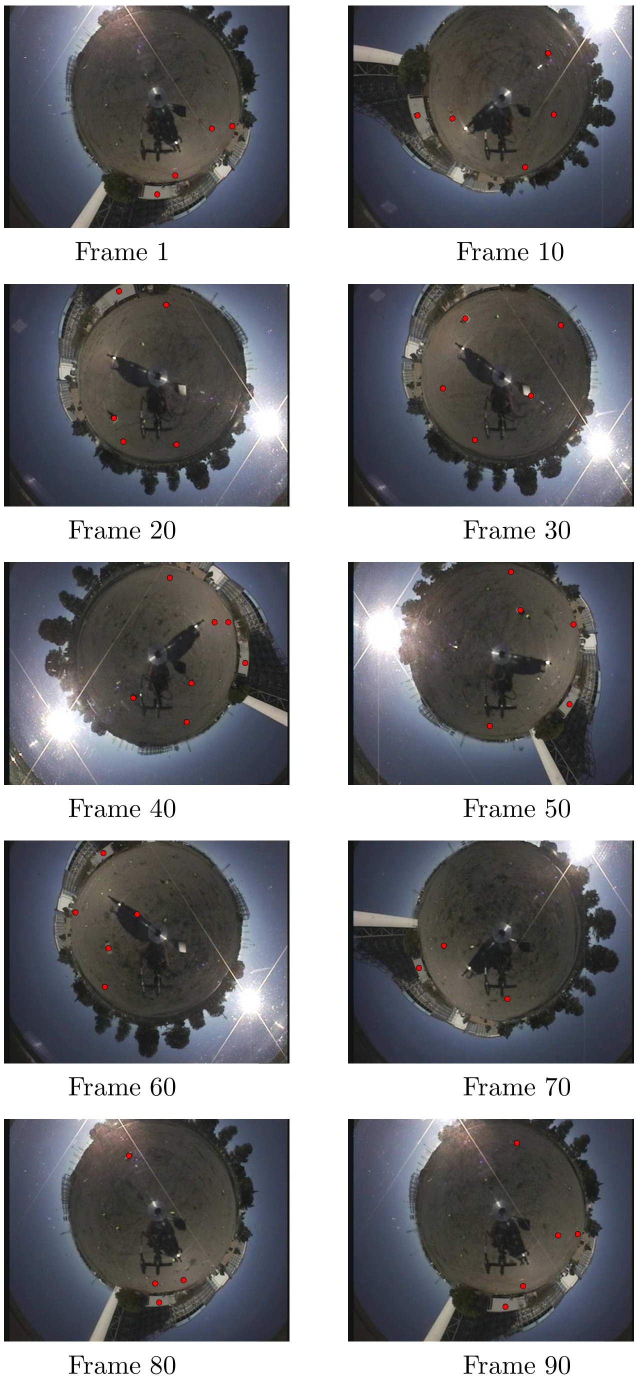


Figure 7.14: Images from the grommit sequence, with tracked features.

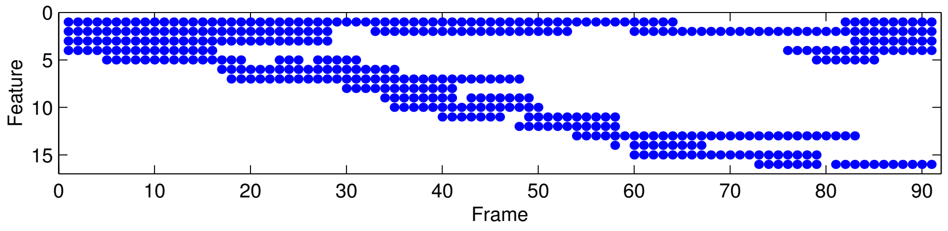


Figure 7.15: Sparseness pattern of bearing measurements for the grommit data.

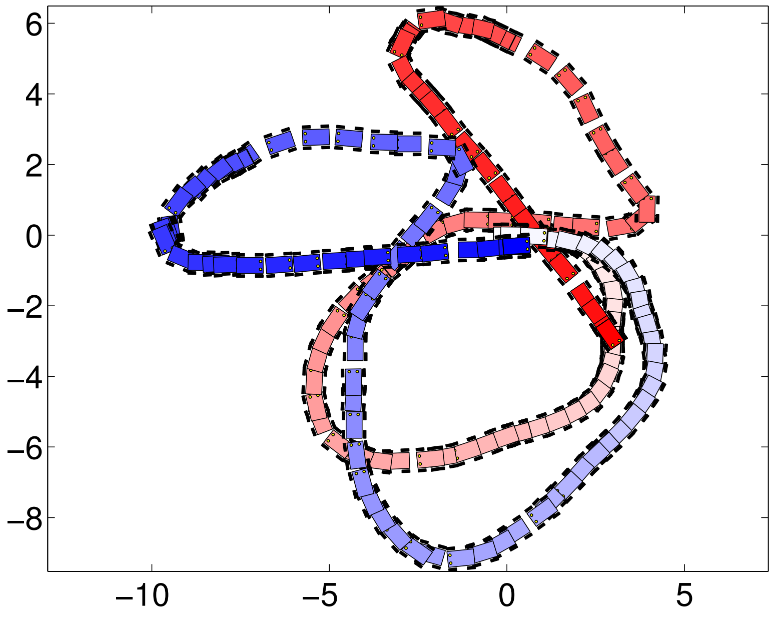


Figure 7.16: Batch estimated trajectory vs. dead reckoning for grommit.

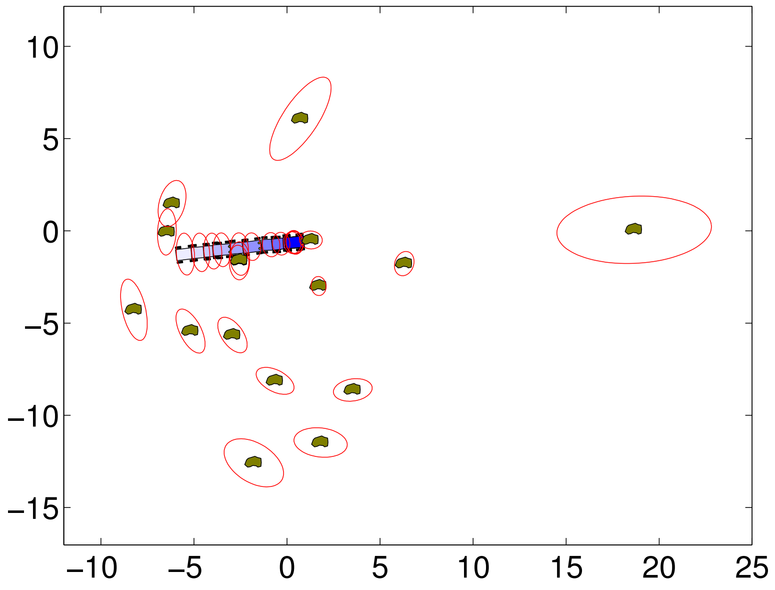


Figure 7.17: Final trajectory segment and map for grommit data estimated using the new algorithm.

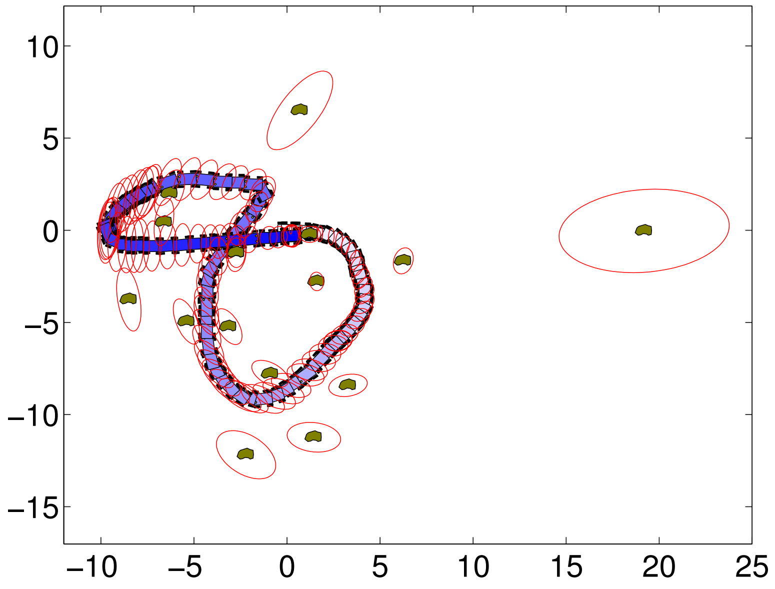


Figure 7.18: Batch estimated trajectory and map for grommit.

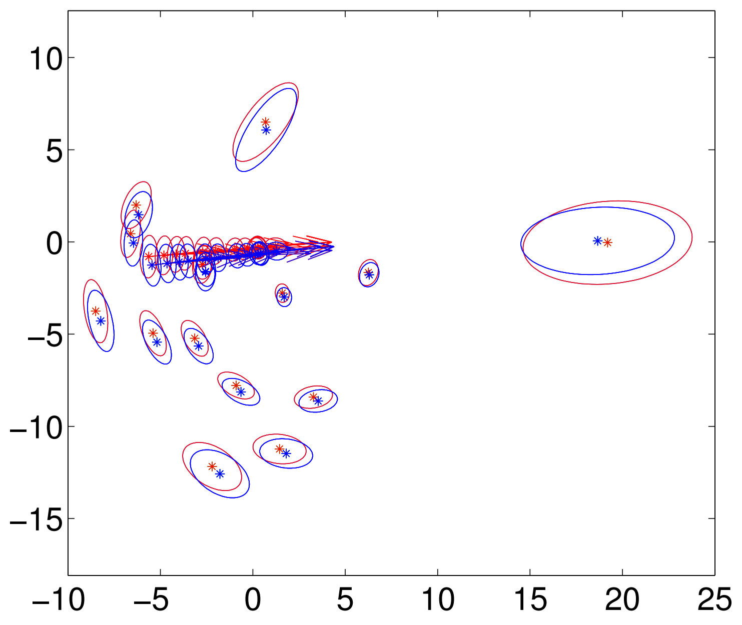


Figure 7.19: Final trajectory segment and map for grommit data estimated using the new algorithm along with the batch result.

as two different features in the data. The correspondence was not fixed, so the information was not taken into account in the mapping algorithm. The batch processing produced estimates for the position of feature 11 and 16 of $(-2.7247, -1.2268)$ and $(-2.7254, -1.2178)$, respectively. This represents a distance of only 91mm. The nonlinear smoothing algorithm estimated the feature locations to be $(-2.5591, -1.6250)$ and $(-2.5574, -1.5961)$, respectively. The nonlinear smoothing algorithm maps the feature to within 2.90 cm. The distance between the absolute position for feature 11 in the batch estimate and smoothing estimate is about 43cm, and the distance between the absolute position for feature 16 in the two estimates is about 41cm. This means that the feature is mapped very consistently relative to the coordinate frame even if the absolute location of the feature is not estimated quite as well.

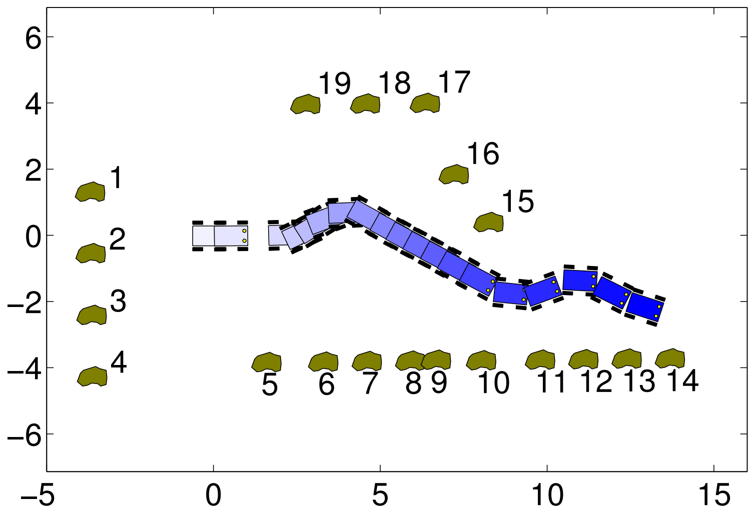


Figure 7.20: Ground truth for the Pioneer test.

7.2.2 Pioneer DX2

The second experiment with real data uses an ActivMedia Pioneer DX2. The robot was driven in the parking lot between Smith Hall and Hamburg Hall on the campus of Carnegie Mellon University. This experiment involves a short traverse with odometry and images being logged. The images were taken at 20 different points along the trajectory while the robot was stopped. Each time that the robot stopped, the pose of the vehicle was measured accurately with a total station. In addition, the locations of 24 features in the environment were measured using the same total station.

The camera on the pioneer was not mounted absolutely straight with respect to the robot body. First attempts at processing the data from the camera either diverged or contained errors that were attributable to bias, e.g. the rover orientation error was significantly biased. In order to overcome this issue, the camera had to be calibrated with respect to the robot body. Because the approach used in this work is 2-dimensional, only the azimuth angle was calibrated. In order to calibrate the azimuth angle, an extra parameter was inserted into the model. This parameter corresponds to the orientation of the camera coordinate frame with respect to the robot coordinate frame. The parameter was optimized by searching over the camera offset which produced the smallest residual error when fitting the entire dataset to the underlying model using the total least squares method discussed in Chapter 3. The minimum residual occurred for an angular offset of 8.1^o.

Figure 7.20 shows the ground truth robot position and feature positions. The

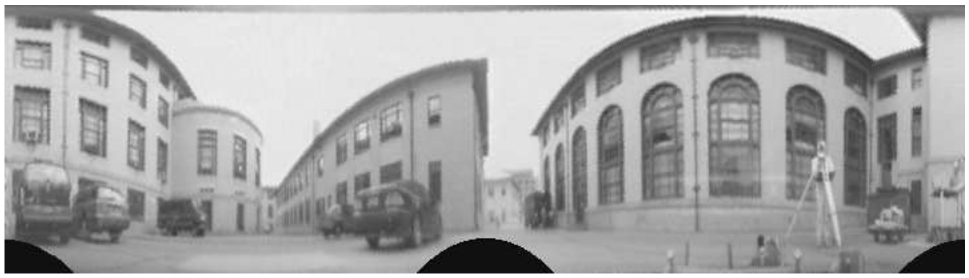


Figure 7.21: First image from pioneer sequence

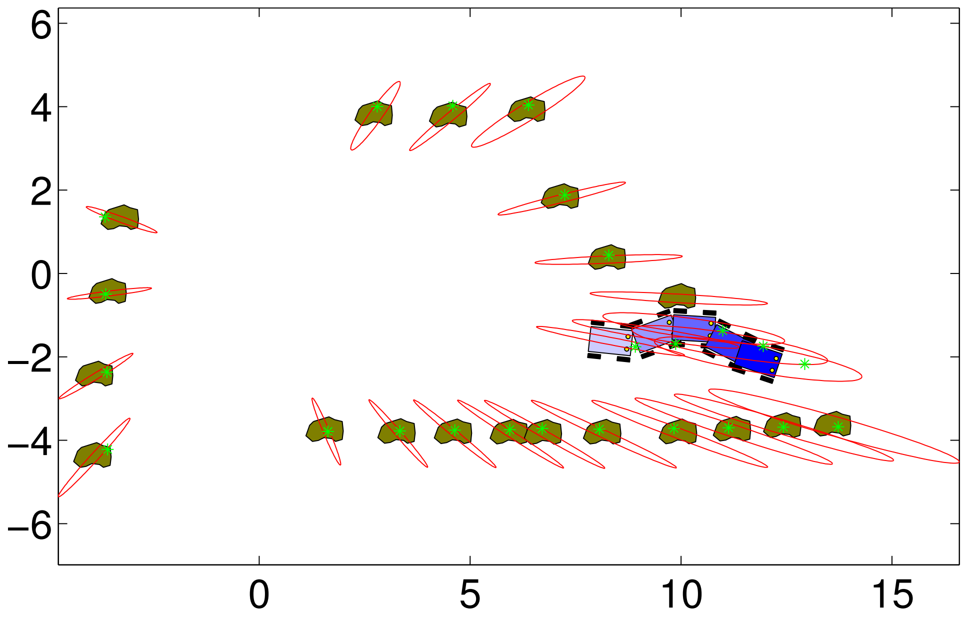


Figure 7.22: Final map for pioneer test.

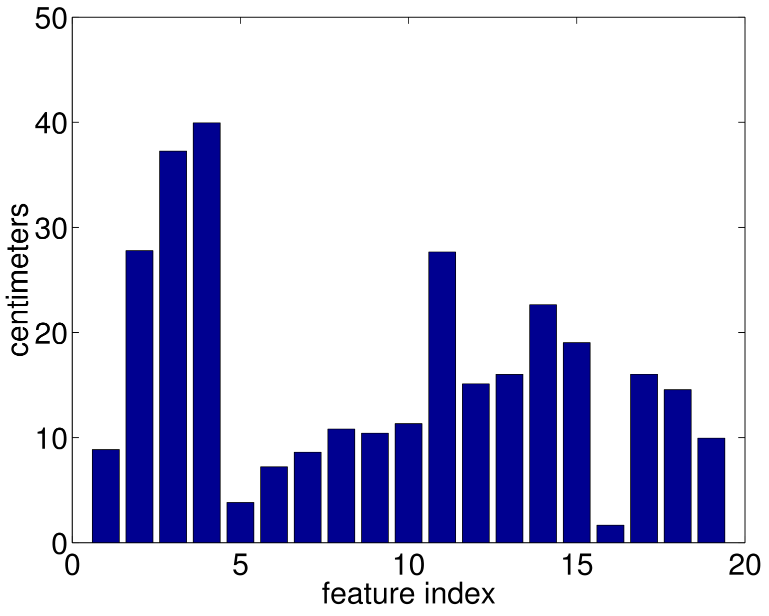


Figure 7.23: Mapping errors for each feature in the pioneer test.

features are corners of the ground floor windows on Hamburg and Smith Hall. The buildings can be seen in an unwarped omnidirectional image in Figure 7.21, which is the first image in the sequence. Figure 7.22 shows the resulting map and final part of the trajectory. The green dots indicate the ground truth locations for the robot and features. The dots are all within the uncertainty ellipses. Figure 7.23 shows the distance between the mapped points and the ground truth. Features 1-4 appear along the wall behind the robot's starting position. These features are observed from positions which do not span much length parallel to the wall. The uncertainty ellipses for these features are elongated and narrow, resulting from the large number of measurements which come from the same direction. These features also have the highest reconstruction error. Most other features are observed from a wider baseline of positions, resulting in uncertainty ellipses which are less oblate and lower reconstruction error.

Chapter 8

Discussion

8.1 Conclusions

This thesis presents several insights into the bearing only SLAM problem. Bundle adjustment and projective factorization techniques from Structure from Motion have been modified to solve the problem, and particularly the use of bundle adjustment provides insight into the nature of the SLAM problem as well as a "gold standard" against which other methods can be tested.

The principle contribution of this thesis is the development of a recursive smoothing algorithm which builds on the Variable State Dimension Filter of McLauchlan and on the derivative free state estimation techniques of Julier, Uhlmann, and Norgaard. The technique starts with the goal of computing the best approximation to the true posterior distribution over state space given the nonlinear measurements. The approximation is done by computing a statistical linearization which can be solved using deterministic sampling from Gaussian quadrature along with importance weighting from Monte Carlo. The full interpolation rule requires a number of samples which is exponential in the number of dimensions. The Unscented filter and Divided Difference filter solve this problem by ignoring all cross terms in the sampling rule, keeping only a minimal set of points which lie along a set of vectors which span the state space. The approach taken here to keep the interpolation tractable is to take advantage of conditional independences and gauge freedoms to reduce the number of dimensions over which the linearization must be computed, then use the full (exponential) sampling rule over a much smaller subspace. The resulting linearization is still exact, without using the simplification from the Unscented or Divided Difference filters which ignore the interactions of different dimensions. Finally, a robust version of the recursive smoothing algorithm is formulated by developing a recursive version of iteratively reweighted least squares (IRLS), which down-weights measurements which cannot be modelled well, rather than a recursive version of total least squares (TLS), which can diverge in the presence of a small number of outliers.

The algorithms developed have been tested in simulation and with some real world experiments to show their consistency, accuracy, and robustness.

At a recent workshop on localization and mapping[19], some criteria were proposed to serve as a basis for comparison for all SLAM algorithms. The first criteria are consistency, convergence, complexity. Later two more criteria, optimality and reliability, were added during discussions.

8.1.1 Consistency

Consistency requires that the error in the state estimate provided by the algorithm is commensurate with the algorithm's own estimate of the error. The algorithm presented in this thesis is empirically consistent, shown by the experiments in Chapter 7. The 99% confidence intervals reported by the filter contain most of the recovered map and trajectory estimates. Furthermore, the method is empirically more consistent than the VSDF for this problem, since the experimental results obtained using the Taylor expansion less often fall within this high probability region.

8.1.2 Convergence

Convergence requires that the error in the state estimate decreases in expectation. In other words, convergence requires estimates which are unbiased and which decrease in variance over time. Unfortunately, the algorithm presented cannot be guaranteed to be unbiased for the same reason that the EKF cannot. The linear approximations to the nonlinearities in the problem can always cause a bias, although the errors in linearization are reduced when smoothing and statistical linearization are used. The variance of the state estimate will decrease over time. The performance of the filter in terms of bias and variance will improve as the time lag increases. As the algorithm becomes closer to full bundle adjustment, the results approach the maximum likelihood result, in which case the algorithm achieves the Cramer-Rao lower bound.

8.1.3 Complexity

Complexity refers to the computational complexity of the algorithm. The algorithm in this thesis is an $O(N^2)$ algorithm where N is the number of landmarks. This is the same complexity as methods based on the EKF. There are a new class of algorithms which use Rao-Blackwellized particle filters and report complexities of $O(N)$ or $O(\log(N))$ but have not yet been used for bearing only SLAM.

8.1.4 Optimality

Optimality requires that the algorithm can provide the lowest variance estimate achievable using the information available. As mentioned above, the algorithm in this thesis should achieve the CRLB if the time lag goes to infinity. With a finite time lag the filter should outperform the EKF since it incorporates more data, and with a time lag of 1 the filter should perform exactly as an IEKF.

8.1.5 Reliability

And finally reliability requires that the algorithm can know when it is failing and report failure. This is done in two ways. The linearization criterion can be used to determine when the nonlinear model is accurately approximated by a linear approximation. This criterion can also be used to dynamically set the time lag, but at the very least can be used to determine whether the expected squared error between nonlinear model and linear approximation is tolerable. If the Taylor expansion is used in an EKF or VSDF framework without analysing higher order terms then there is no guarantee that the approximation is valid. Secondly, since a finite time history of measurements is available, a χ^2 test for significance can be made to check that the model recovered by the algorithm matches the measurements to a reasonable degree. An excessively large χ^2 value can imply filter divergence or the presence of outliers in the data, and in either case should raise a red flag in terms of reliability.

8.2 Comparison with existing methods

This section will briefly present some qualitative comparisons with existing SLAM techniques. Not all of these techniques have been applied to bearing only SLAM.

8.2.1 Bundle Adjustment

Bundle adjustment is optimal in the sense that the objective function, the total squared error between the measurements observed and those that are predicted by the sensor models, is minimized directly. This is often referred to in computer vision as the *reconstruction error*. Barring annoyances such as local minima, bundle adjustment should achieve the lowest possible reconstruction error. However, this optimum is achieved at a high computational cost.

Nonlinear recursive smoothing attempts to approximate full bundle adjustment with a recursive algorithm. In some sense, this approximation offers a full range of algorithms which mix Kalman filtering and bundle adjustment by providing a range of time lags that can be used for filtering. A one-step filter is equivalent to the EKF, and an infinite memory nonlinear optimization is equivalent to bundle adjustment. A graphical representation is shown in Figure 8.1.

8.2.2 Kalman Filters

The extended Kalman filter certainly has a long and rich history in the SLAM community[79, 62, 80, 47, 20, 29]. The *stochastic mapping* algorithm of Smith and Cheeseman[79], which is essentially an extended Kalman filter, has been

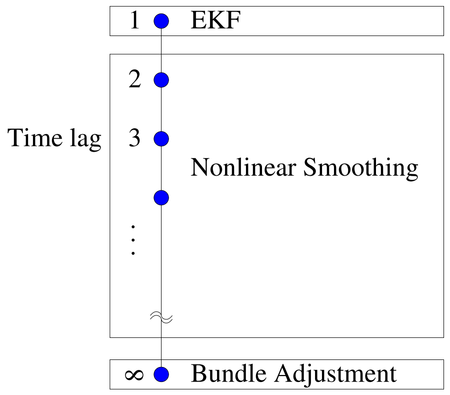


Figure 8.1: Relationship between nonlinear smoothing and both the EKF and bundle adjustment.

used and referenced extensively. The filtering equations have even been used to analyze the nature of the problem itself[30, 31]. The extended Kalman filter is the foundation for both the unscented filter, which uses deterministic sampling for computing posterior statistics, and for the Variable State Dimension Filter, which uses nonlinear smoothing rather than one-step filtering. The method outlined in this thesis uses deterministic sampling to compute posterior statistics for nonlinear smoothing, relying on the UKF and VSDF. For this reason, it makes sense to directly consider the relationship between the new method and both the UKF and VSDF below.

8.2.3 Unscented filter

The method described in this thesis is related to the unscented filter in its use of deterministic sampling for computing posterior statistics. However, there are three major differences between the algorithms.

The first difference is that the UKF does filtering and the method described here uses smoothing. In a UKF, at each sensor update the measurement is incorporated directly into the state estimate and posterior statistics are computed for the data. The method described in this thesis uses smoothing to incorporate data from multiple time steps, so the state estimates should be more accurate before posterior statistics are computed.

The other difference is that the unscented filter computes the first and second moments of the posterior directly, whereas the method described here computes the best linear fit for the motion and measurement model, then computes the posterior statistics as an analytic function of the linearized model. This is essentially the difference between the UKF and DD1 filters, discussed in Chapter 5, and one of the interesting proposals for future work is to compare moment matching and statistical linearization for bearing only SLAM.

Finally, the UKF performs multidimensional moment matching by embedding a one-dimensional sampling scheme along the principal axes of the posterior distribution in state space. This potentially misses nonlinaer interactions between the different dimensions. The method used in this thesis does a full multidimensional quadrature by reducing the dimensionality of the integrals to be computed.

8.2.4 Variable State Dimension Filter

The Variable State Dimension Filter is a nonlinear smoothing algorithm which was developed in the context of SFM. The method uses the same linearization technique as the EKF for computing measurement updates and posterior statistics. The method in this thesis extends the VSDF to use deterministic sampling to compute posterior statistics. The extension is similar to the relationship that the UKF has to the EKF. In terms of computational complexity the method is the same as the VSDF. When the problem has significant nonlinearities the new method can be expected to outperform the VSDF, and for purely linear models the two should perform the same, since the linearizations computed numerically should exactly match the analytic derivatives. The experimental results presented here explicitly compare the difference in performance between the VSDF and the new technique for bearing only SLAM.

8.2.5 Hierarchical mapping

Chong and Kleeman presented a technique for breaking an environment into small local maps which can be mapped separately[16]. Decoupled stochastic mapping[49] is a similar map management strategy, and the ATLAS framework[7] is the latest generation of this technique which adds topological inference and global map generation. These techniques all require a mapping algorithm to build the local submaps. Typically an EKF style algorithm is used. The method presented in this thesis could be used in place of the EKF in the local map building.

8.2.6 Filtering with invariants

The relative filter[22], GPF[66], and invariant filter[25] recursively estimate invariant relationships between landmark features, and then recover a global map from the invariant relationships. Existing approaches compute one step filter estimates of the invariant relationships. The nonlinear smoothing approach taken here could be used to estimate invariants as well, replacing the nonlinear filtering with nonlinear smoothing. The principle difficulty is in the strategies for avoiding correlation between invariants. Accounting for correlations in the bearing only case becomes difficult if multiple invariants were estimated using measurements that are correlated by virtue of sharing common motion information.

8.2.7 Delayed Decision Making

Delayed decision making[48, 75] is closely related to the method presented in this thesis. Leonard and Rikoski are interested in the bearing only SLAM problem, particularly in the context of underwater robots. They have introduced the terminology *partially observable* to describe measurements which cannot uniquely determine the state of a feature, and have developed a strategy for initializing and updating features with measurements from multiple time steps, similarly to the work presented here. However, the measurements used to initialize the feature are not used for the update, and vice versa. The algorithm in this thesis uses all available measurements to initialize new features, obtaining an "optimal" estimate relative to the reconstruction error with respect to available measurements. Furthermore, the posterior statistics are computed only for the oldest measurement in the filter, leaving other measurements as nonlinear quantities until they have been in the filter for the specified lag time. The delayed decision making algorithm performs a *batch update* which updates the state with respect to a delayed time window of measurements using the Kalman filtering update equations and then clears out the old measurements. The posterior statistics are computed using a Taylor expansion, unlike the method described here which uses deterministic sampling to find the best local linear approximation.

8.2.8 Particle Filters

The two primary difficulties for particle filters for SLAM are the high dimensionality of the state space and the mix of static and dynamic states within the problem. Particle filters have difficulties with state estimation in high dimensions. Drawing samples from a reasonable prior or directly from the posterior can be difficult. In Structure from Motion, this has been dealt with by directly solving two frame SFM problems and sampling from the results[12] in a manner similar to bootstrap[15]. Using nonparametric, sample based density estimates, particle filters can represent arbitrary posterior distributions. The method described in this thesis assumes a normal density for the posterior and seeks the best approximation from this family. For multimodal or otherwise highly non-Gaussian posteriors, particle filters can be expected to perform well, but if the posterior is approximately Gaussian, then assuming a parametric form will perform better.

Stochastic time-evolving state models can be tracked well with particle filters because the stochastic nature of the problem helps the filter recover from sample impoverishment by diffusing samples which come from the same parent. However, the map is often modeled as a static state in SLAM. Once a sample containing a particular map hypothesis is eliminated, that map hypothesis is lost along with it. Some *ad hoc* approaches have been suggested for dealing with this issue[11], but as yet there is not a compelling sample based parameter estimation technique. An alternative is to mix representations using Rao-Blackwellization.

8.2.9 Rao-Blackwellized Particle Filters

The FastSLAM algorithm of Thrun and Montemerlo[59] is a Rao-Blackwellized particle filter. The filter samples over part of the state space, the robot motion, and computes analytic density estimates over the other, the map. The analytic portion of the filter is implemented using Kalman filters, and due to the structure of the problem, this partioning also has the advantage that the mapping can be decoupled so that each of the landmarks is mapped independently, i.e. each Kalman filter is of a fixed size. The partioning is exact, and the resulting algorithm is $O(N)$ for N landmarks, or perhaps $O(\log(N))$[59] using a tree representation for the maps. This represents a significant improvement in computational complexity over the EKF, UKF, VSDF, and the method discussed here.

However, similar to the EKF, FastSLAM uses only a one step update, and to date FastSLAM has only been used for sensors which measure bearing and range together. An interesting proposal for future work would be to tackle bearing-only SLAM with an algorithm which incorporates the mixed parametric/nonparametric approach and partitioning in a smoothing context to handle bearing only or range only measurements. The resulting algorithm should benefit from the reduction in computational complexity from Rao-Blackwellization and gain in filter accuracy from smoothing.

8.2.10 Linear subspace methods

The trilinear tensor has been used for SFM[33] and localization and mapping[27], and projective factorization has been used for SFM[83, 52, 53] and could be used for bearing only SLAM as well. The methods have the advantage that because the problem is expressed as a linear one, convergence is less often hindered by local minima. This means that in practice the algorithms do not require good initial guesses to converge to a reasonable solution.

However, neither of the approaches takes all of the information from odometry into consideration. It is not convenient to express the odometry measurement in the same framework as the trilinear tensor constraint or the SVD factorization. Furthermore, in SFM these approaches are often followed by bundle adjustment in order to obtain the best possible results. The nonlinear smoothing algorithm approximates bundle adjustment directly.

8.2.11 Scan matching and correlation

Scan matching[50] and correlation techniques[39] have been used recently for SLAM and for SLAM with moving object detection and removal[96]. These methods work well with range sensors, but with a bearing only sensor or monocular camera, scan matching is not applicable. The most direct analogy to scan matching for projective geometry are methods based on linear subspaces[33, 27, 83, 52, 53]

8.2.12 Grid based methods

Grid based methods have been used for mapping for a long time[60] and have the advantage that they do not require a feature based map. However the maps made using grid based methods typically record the probability of occupancy for each cell in the grid marginalized over other unknowns such as robot position. Because of this, grid based methods do not track the full joint posterior between the robot and map or even the joint probability between neighboring cells. This is understandable since the state space is intractable, but it can have serious consequences since knowledge of one part of the map can have a strong affect on the robot's knowledge of other parts of the map[30]. As with the popular Kalman filter based methods, the nonlinear smoothing algorithm presented here represents the environment in a feature based map, and maintains a full joint posterior over state space.

8.3 Future work

One important goal for sampling based methods in recursive estimation is to eliminate the need for analytic Jacobians. The method developed in this thesis relies on analytic derivatives for optimization but uses statistical linearization to compute posterior statistics. It should be possible to extend this to a fully sample based method which no longer requires analytic differentiation. There are several advantages to doing this. The first is that some estimation problems do not have differentiable models. The second is that Jacobians can be tricky to compute and implement correctly. Finally, a turnkey filter implementation tool[9] would be easier if a user can simply specify the models rather than also providing Jacobians.

Minka's thesis[58] presents a method called expectation propagation (EP) for approximating densities in arbitrary Bayes networks which might be useful for the problem considered here. In EP, the moments of the posterior distribution are iteratively estimated directly. Moment matching has been used in recursive filtering before, and is more akin to the unscented filter than the method described in this thesis. It will be interesting to compare the two methods.

Data association is a crucial issue in SLAM, and has gone largely unaddressed in this dissertation. The data association problem is one which is currently gaining popularity in the SLAM community. This is partly because the community is beginning to settle on a set of algorithms which behave well and are well understood, and partly because as SLAM implementations move out of the lab and into the world, data association is an issue which can become very complex very quickly.

In image based target tracking, feature correspondence is often established in measurement space, i.e. on the image plane. By combining appearance cues and motion cues on the image plane, features can be tracked as images become available. Images provide a richness of data which partly makes up for the loss of range information by returning enough data to capture the appearance of a feature. Cues such as color, texture, shape and size can be used to establish correspondence, or a straightforward template comparison can be used when appearance constancy is a reasonable assumption. Furthermore, tracking features and establishing correspondence on the image plane does not require good estimates for the positions of features in the world or for good estimates of the camera (rover) position. However, in order to properly handle features which leave the field of view and return, and likewise handle mapping a large cyclic environment, some recourse to state space is necessary. In the example with NASA Ames' grommit rover, two features were mapped independently even though they were actually the same feature. The first time the feature was encountered was near the starting location, and the feature left the field of view. When the feature returned to the field of view it was reinitialized and tracked. Ideally, the algorithm would be aware of the fact that the two features are located at almost the same location (especially given the Mahalanobis distance with respect to the estimated mapping accuracy) and that they are not simultaneously visible. Furthermore, a check for appearance consistency might have shown a good match.

Outside of the context of metric based mapping and feature based measurements, there is some interest in topological SLAM and visual cues for localization and mapping which go beyond extracting a few features from an image. A 640×480 image contains 307,200 pixels. If a handful of features are extracted from the image and each feature corresponds to a small window of neighboring pixels, most of the data collected is ignored. Some alternate approaches such as color histogramming have proved useful for localization[93]. Moving to environments which do not have easily identifiable point features and do not lend themselves to feature based metric mapping might be solveable using topological methods and matching techniques which use color histograms, textures, or other cues.

Bibliography

[1] A. Adam, E. Rivlin, and H. Rotstein. Fusion of fixation and odometry for vehicle navigation. In *Proceedings of International Conference on Robotics and Automation*, volume 2, pages 1638–1643, 1999.

[2] A. Adam, E. Rivlin, and H. Rotstein. Fusion of fixation and odometry for vehicle navigation. *IEEE Transactions on Systems, Man, and Cybernetics*, 29:593–603, November 1999.

[3] M. Athans, R. P. Wishner, and A. Bertolini. Suboptimal state estimation for continuous-time nonlinear systems from discrete noisy measurements. In *IEEE Transactions on Automatic Control*, volume 13, pages 504–518, October 1968.

[4] Ali Azarbayejani and Alex P. Pentland. Recursive estimation of motion, structure and focal length. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6):562–575, 1995.

[5] R. Benjemaa and F. Schmitt. A solution for the registration of multiple 3d point sets using unit quaternions. In *Proceedings of the European Conference on Computer Vision*, volume 2, pages 34–50, 1998.

[6] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[7] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller. An atlas framework for scalable mapping. In *Proceedings of IEEE ICRA*, 2003.

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

[9] W. Buntine and et al. Transformation systems at nasa ames. In *Proceedings of Workshop on Software Transformation Systems*, May 17th 1999.

[10] W. Burgard, D. Fox, H. Jans, C. Matenar, and S. Thrun. Sonar-based mapping with mobile robots using EM. In *Proc. of the International Conference on Machine Learning*, 1999.

[11] J. Carpenter, P. Clifford, and P. Fearnhead. Improved particle filter for non-linear problems. *IEE Proceedings-Radar, Sonar and Navigation*, 146(1):2–7, 1999.

[12] P. Chang and M. Hebert. Robust tracking and structure from motion through sampling based uncertainty representation. In *Proceedings of the International Conference on Robotics and Automation*, May 2002.

[13] R. Chen and J. S. Liu. Mixture kalman filters. *Journal of the Royal Statistical Society, Series B*, 62(3):493–508, 2000.

[14] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, 1992.

[15] K. Cho, P. Meer, and J. Cabrera. Performance assessment through bootstrap. *IEEE Trans. PAMI*, pages 1185–1198, 1997.

[16] K. Chong and L. Kleeman. Large scale sonarray mapping using multiple connected local maps. In *International Conference on Field and Service Robotics*, pages 278–285, 1997.

[17] K.S. Chong and L. Kleeman. Indoor exploration using sonar sensor array: A dual representation strategy. In *Proceedings of the International Conference on Intelligent Robots and Systems*, volume 2, pages 676–82, 1997.

[18] K.S. Chong and L. Kleeman. Sonar based map building for a mobile robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1700–5, 1997.

[19] H. I. Christensen, editor. *Proceedings of the SLAM Summer School*. KTH, Stockholm, Sweden, July 2002.

[20] S. Clark and G. Dissanayake. Simultaneous localisation and map building using millimetre wave radar to extract natural features. In *Proc. IEEE ICRA*, pages 1316–1321, 1999.

[21] S. Clark and H. Durrant-Whyte. Autonomous land vehicle navigation using millimetre wave radar. In *Proceedings of ICRA*, pages 3697–3702, 1998.

[22] M. Csorba and H. Durrant-Whyte. A new approach to map building using relative position estimates. *SPIE Vol. 3087*, pages 115–125, 1997.

[23] Michael Csorba. *Simultaneous Localization and Map Building*. PhD thesis, Department of Engineering Science, University of Oxford, 1997.

[24] M. Deans and M. Hebert. Experimental comparison of techniques for localization and mapping using a bearings only sensor. In *Proc. of the ISER '00 Seventh International Symposium on Experimental Robotics*, December 2000.

[25] M. Deans and M. Hebert. Invariant filtering for simultaneous localization and mapping. In *Proceedings of IEEE ICRA*, pages 1042–7, April 2000.

[26] F. Dellaert, S. Seitz, C. Thorpe, and S. Thrun. Structure from motion without correspondence. Technical Report CMU-RI-TR-99-44, Robotics Institute, Carnegie Mellon University, December 1999.

[27] F. Dellaert and A. Stroupe. Linear 2d localization and mapping for single and multiple robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, May 2002.

[28] G. Dissanayake, H. Durrant-Whyte, and T. Bailey. A computationally efficient solution to the simultaneous localisation and map building problem, 2000.

[29] M.W.M.G Dissanayake and et al. An experimental and theoretical investigation into simultaneous localization and map building (slam). In *Proc. 6th International Symposium on Experimental Robotics*, pages 171–180, 1999.

[30] M.W.M.G. Dissanayake and et al. A solution to the simultaneous localization and map building (slam) problem. Technical Report ACFR-TR-01-99, Australian Centre for Field Robotics, University of Sydney, Australia, 1999.

[31] M.W.M.G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, June 2001.

[32] O. D. Faugeras, L. Quan, and P. Sturm. Self-calibration of a 1d projective camera and its application to the self-calibration of a 2d projective camera. In *Proceedings of ECCV*, pages 36–52, 1998.

[33] O. D. Faugeras, L. Quan, and P. Sturm. Self-calibration of a 1d projective camera and its application to the self-calibration of a 2d projective camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:1179 –1185, October 2000.

[34] H. Feder, J. Leonard, and C. Smith. Adaptive mobile robot navigation and mapping. *International Journal of Robotics Research*, 18(7):650–668, July 1999.

[35] M.A. Fischler and R.C. Bolles. Ransac random sample concensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 26:381–395, 1981.

[36] Andrew W. Fitzgibbon and Andrew Zisserman. Automatic camera recovery for closed or open image sequences. In *Proceedings of the European Conference on Computer Vision*, pages 311–326, 1998.

[37] S. Gold, A. Rangarajan, C. Lu, S. Pappu, and E. Mjolsness. New algorithms for 2d and 3d point matching: pose estimation and correspondence. *Pattern Recognition*, 31(8):1019–1031, 1998.

[38] J. Goldberger. Registration of multiple point sets using the em algorithm. In *Proceedings of the International Conference on Computer Vision*, volume 2, pages 730–736, 1999.

[39] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proc. IEEE Symp. CIRA*, pages 318–325, 1999.

[40] Richard I. Hartley. Euclidean reconstruction from uncalibrated views. In Zisserman Mundy and Forsyth, editors, *Applications of Invariance in Computer Vision*, pages 237–256. Springer Verlag, 1994.

[41] B. K. P. Horn. Closed-form solution of absolute orientation using unit quatemions. *Journal of the Optical Society of America*, 4:629–642, 1987.

[42] P. J. Huber. *Robust Statistics.* John Wiley & Sons, New York, 1981.

[43] K. Ito and K. Xiong. Gaussian filters for nonlinear filtering problems. *IEEE Transactions on Automatic Control*, 45(5):910–927, May 2000.

[44] S. Julier, J. Uhlmann, and H. Durrant-Whyte. A new approach for filtering nonlinear systems. In *Proceedings of the 1995 American Controls Conference*, pages 1628–1632, 1995.

[45] S. Julier and J. K. Uhlmann. Reduced sigma point filters for the propagation of means and covariances through nonlinear transformations, 2001.

[46] S. J. Julier and J. K. Uhlmann. A general method for approximating nonlinear transformations of probability distributions. Technical report, Dept. of Engineering Sciences, University of Oxford, 1996.

[47] J. J. Leonard and H. F Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. In *IEEE/RSJ International Workshop on Intelligent Robots and Systems IROS '91*, pages 1442–1447, 1991.

[48] J. J. Leonard and R. Rikoski. Incorporation of delayed decision making into stochastic mapping. In D. Rus and S. Singh, editors, *Experimental Robotics VII.* Springer-Verlag, 2001.

[49] John J. Leonard and Hans Jabob S. Feder. Decoupled stochastic mapping. Technical Report 99-1, MIT Marine Robotics Laboratory, Cambridge, MA 02139, USA, 1999.

[50] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4(4):333–349, 1997.

[51] D. J. C. MacKay. Introduction to monte carlo methods. In M. I. Jordan, editor, *Learning in Graphical Models*. MIT Press, 1999.

[52] S. Mahamud and M. Hebert. Iterative projective structure from multiple views. In *poster session at CVPR2000*, pages II:430–7, 2000.

[53] S. Mahamud, M. Hebert, Y. Omori, and J. Ponce. Provably convergent iterative methods for projective structure from motion. In *CVPR 2001*, pages I:1018–25, 2001.

[54] P. Maybeck. *Stochastic Models, Estimation, and Control*. Academic Press, Inc., 1979.

[55] P. McLauchlan. A batch/recursive algorithm for 3d scene reconstruction. In *Proceedings of CVPR 2000*, pages 738–743, 2000.

[56] Philip F. McLauchlan. Gauge invariance in projective 3d reconstruction. In *Proceedings IEEE Workshop on Multi-View Modeling and Analysis of Visual Scenes (MVIEW'99)*, pages 37–44, 1999.

[57] 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 GU2 5XH, 1999.

[58] T. P. Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

[59] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. Fastslam: A factored solution to the simultaneous localization and mapping problem. *To appear in Proceedings of AAAI*, 2002.

[60] H. Moravec and A. Elfes. High resolution maps from angle sonar. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 116–121, 1985.

[61] Daniel Morris. *Gauge invariance in structure from motion*. PhD thesis, Carnegie Mellon University, 2001.

[62] P. Moutarlier and R. Chatila. Stochastic multisensory data fusion for mobile robot location and environment modelling. In *5th Int. Symposium on Robotics Research*, 1989.

[63] S. Nayar. Catadioptric omnidirectional camera. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1997.

[64] P. Neugebauer. Geometrical cloning of 3d objects via simultaneous registration of multiple views. In *Proceedings of the International Conference on Shape Modelling and Applications*, pages 130–9, March 1997.

[65] P. M. Newman and H. Durrant-Whyte. A new solution to the simultaneous localisation and map building (slam) problem - the gpf. Technical report, Australian Centre for Field Robotics, University of Sydney, 2000.

[66] Paul Michael Newman. *On the Structure and Solution of the Simultaneous Localization and Mapping Problem*. PhD thesis, Australian Centre for Field Robotics, University of Sydney, 1999.

[67] M. Nørgaard. Kalmtool toolbox, matlab toolbox. *http://www.iau.dtu.dk/research/control/kalmtool.html*.

[68] M Nørgaard, N. K. Poulsen, and O. Ravn. Advances in derivative-free state estimation for nonlinear systems. Technical Report IMM-REP-1998-15, Technical University of Denmark, 2800 Lyngby, Denmark, 2000.

[69] J. Oliensis and J. Inigo Thomas. Incorporating motion error in multi-frame structure from motion. In *IEEE Workshop on Visual Motion*, pages 8–13, 1991.

[70] C. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. Technical Report CMU-CS-93-219, Carnegie Mellon University, 1993.

[71] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C*. Cambridge University Press, 1988.

[72] Kari Pulli. Multiview registration for large data sets. In *Int.Conf. on 3D Digital Imaging and Modeling*, pages 160–168, 1999.

[73] G. Qian, R. Chellapa, Q. Zheng, and J. Ortolf. Camera motion estimation using monocular image sequences and inertial data. Technical Report CAR-TR-909, Center for Automation Research, University of Maryland, 1999.

[74] Donald B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, AC-24(6), December 1979.

[75] R. J. Rikoski, J. J. Leonard, and P. M. Newman. Stochastic mapping frameworks. In *Proceedings of IEEE ICRA*, 2002.

[76] A. Shashua. Trilinear tensor: The fundamental construct of multiple-view geometry and its applications. *Lecture Notes in Computer Science*, 1315:190–??, 1997.

[77] H. Shum, K. Ikeuchi, and R. Reddy. Principal component analysis with missing data and its application to object modeling. Technical Report CMU-CS-93-213, School of Computer Science, Carnegie Mellon University, 1993.

[78] Heung-Yeung Shum, Qifa Ke, and Zhengyou Zhang. Efficient bundle adjustment with virtual key frames: A hierarchical approach to multi-frame structure from motion. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, pages 538–543, June 1999.

[79] R. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *The International Journal of Robotics Research*, 5(4), 1986.

[80] Randall Smith, Matthew Self, and Peter Cheeseman. *Estimating Uncertain Spatial Relationships in Robotics*, chapter 3, pages 167–193. Springer-Verlag, 1990.

[81] S. Soatto, P. Perona, R. Frezza, and G. Picci. Recursive motion and structure estimation with complete error characterization. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 428–433, 1993.

[82] A. J. Stoddart and A. Hilton. Registration of multiple point sets. In *Proceedings of ICPR '96*, pages 40–44, 1996.

[83] P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure and motion. In *ECCV96*, pages II:709–720, 1996.

[84] C. J. Taylor, D. J. Kriegman, and P. Anandan. Structure and motion in two dimensions from multiple images: A least squares approach. In *Proceedings of the IEEE Workshop on Visual Motion*, pages 242–248, 1991.

[85] S. Thrun. Simultaneous mapping and localization with sparse extended information filters: Theory and initial results. Technical Report CMU-CS-02-112, School of Computer Science, Carnegie Mellon University, 2002.

[86] S. Thrun, W. Burgard, and D. Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning and Autonomous Robots*, 31(5):1–25, 1998.

[87] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping. In *Proceedings of the International Conference on Robotics and Automation*, pages 321–328, 2000.

[88] C. Tomasi and T. Kanade. Shape and motion from image streams: a factorization method. Technical Report CMU-CS-92-104, Carnegie Mellon University, 1992.

[89] B. Triggs. Factorization methods for projective structure from motion. In *Proceedings of CVPR*, pages 845–851, 1996.

[90] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. In *To appear in Vision Algorithms: Theory & Practice*. Springer-Verlag, 2000.

[91] J. K. Uhlmann. Unscented filter, matlab implementation. *http://www.robots.ox.ac.uk/ siju/work/work.html*.

[92] J. K. Uhlmann, S. J. Julier, and M. Csorba. Nondivergent simultaneous map-building and localization using covariance intersection. In *SPIE Proceedings: Navigation and Control Technologies for Unmanned Systems II*, volume 3087, pages 2–11, 1997.

[93] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *Proceedings of IEEE ICRA*, 2000.

[94] J. R. van Zandt. A more robust unscented filter. Technical report, Mitre Corp., 2001.

[95] E. A. Wan and R. van der Merwe. The unscented kalman filter for nonlinear estimation. In *Proc. of IEEE Symposium 2000 (AS-SPCC)*, Oct. 2000.

[96] C. Wang and C. Thorpe. Localization and mapping with detection and tracking of moving objects. In *Proc. IEEE ICRA*, 2002.

[97] Y. Yagi, K. Shouya, and M. Yachida. Environmental map generation and egomotion estimation in a dynamic environment for an omnidirectional image sensor. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 3493–3498, 2000.

[98] Zhengyou Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. Technical Report No. 2676, INRIA, 1995.