Multi-frame Measurement Fusion for State Estimation

by

David D. Kroetsch

A thesis
presented in fulfillment of the
thesis requirement for the degree of
Master of Applied Science

Department of Mechanical and Mechatronic Engineering
University of Waterloo

Waterloo, Ontario, August 2007

Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

David D. Kroetsch

Abstract

David Kroetsch
University of Waterloo

Master of Applied Science
September 2007

Multi-frame Measurement Fusion for State Estimation

Simultaneous Localization and Mapping (SLAM) is a technique used by robots and autonomous vehicles to build up a map while operating in an unknown environment, while at the same time keeping track of a current position estimate. This process is not as straightforward as it may sound. Due to inherent uncertainties in tracking the robot's motion and observing the world around it, errors accumulate.

Incorporating External Localization (EL) sensors, such as Global Positioning System (GPS) receivers, which are not subject to the same type of drift over time, can allow a system to eliminate errors from the SLAM process. This, however, requires the system to work with measurements from both the SLAM and EL coordinate frames. Since the SLAM coordinate origin is dependent upon the vehicle's starting location, which is unknown, the relationship between it and the EL frame is unknown and must be estimated.

The main contributions of this thesis arise from a novel approach to integrating EL with SLAM, by estimating a transformation between the SLAM and external reference frames. The Constrained Relative Submap Filter (CRSF) SLAM is a computationally efficient SLAM filter operates on a series of local submaps, instead of one large map, as with Extended Kalman Filter (EKF) SLAM. By integrating the transformation estimation process with CRSF SLAM , a method to correct submap locations with EL is presented. This eliminates long term estimation drift, aids in loop closing and allows for accurate map generation with a reference to an external frame.

Acknowledgements

I thank my supervisors Dr. Chris Clark and Dr. David Wang for their guidance throughout this project, and for the freedom they gave me to solve problems in my own time. Dave, thanks for getting the ball rolling and showing me the benefits of continuing my academics as a way to open the door to bigger and better things. Right from the beginning, when I knocked on your office door in 1997, looking to build flying robots, your encouragement and advice has been invaluable.

Chris, thanks for guiding me through the academic aspects of this research, helping me through the politics and encouraging my desire to travel. Your stories got me excited to travel and with your help it turned into something practical and feasible. It was way more than a research trip and really opened my mind, giving me an opportunity to see the whole world first-hand, quite literally.

To Dr. Stefan Williams, thanks for the fantastic opportunity you gave me at the University of Sydney. Being involved in the research your team was conducting was a great experience and having a Canadian connection in the midst of my world tour was refreshing.

I would also like to thank my family. I'd like to thank my parents for believing in me from day one, and giving me the encouragement to follow through. The offers to deliver dinner on those long weekends of thesis writing meant the world! To my sister, thanks for listening to me ramble when I just needed someone to talk to.

Thank you.

Contents

List of Tables

List of Figures

List of Notations

The following detail the symbols and notations used throughout this thesis.

- Left superscripts such as ${}^{S}z$ are used to denote the frame in which a quantity is expressed. The previous example, denotes a measurement z in the **S** frame.

${}^{S}z_s$	a SLAM s measurement represented in the **S** frame
$\mathbf{T}^{\mathbf{E}}_{\mathbf{S}}$	a transformation **T** from the **S**-frame to the **E**-frame

Chapter 1

Introduction

1.1 Objectives

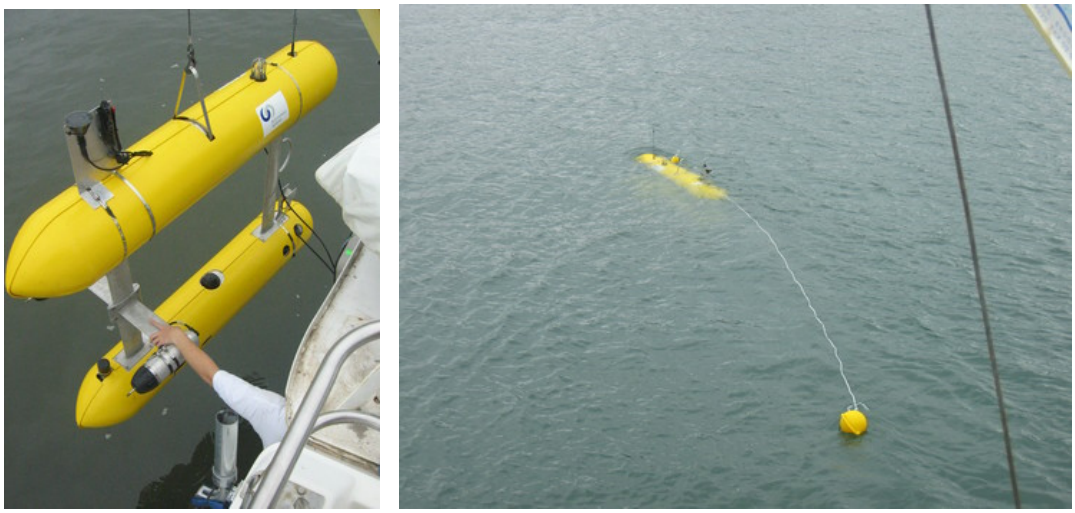
Autonomous robot localization in previously unexplored environments requires the robot to build a map of its surroundings, and keep track of its location within that map. This ability relies on information, which the vehicle gathers with its sensors about its motion and characteristics of the area around it, to build a map through a process known as Simultaneous Localization and Mapping (SLAM).

These robotic systems typically begin with no knowledge of their environment or their location. They must extract some useful metrics or features from the sensor data gathered in order to make relative measurements to nearby landmarks. As the vehicle moves, the relative distances to these landmarks change and the system can estimate the positions of these landmarks and the relative location of the vehicle.

The objective of this thesis is to provide a method to fuse absolute vehicle location measurements, such as Global Positioning System (GPS), which come from outside of the SLAM system, with the relative map and vehicle position estimates which SLAM provides.

1.2 Background and Motivation

There are numerous field robotics systems which are designed to work in unknown environments. The Seabed system running at the Australian Centre for Field Research (ACFR) at the University of Sydney is one example of an autonomous underwater vehicle. Photographs of this vehicle are shown in Figure 1.1. Equipped with many sensors to detect its motion, SLAM can be used by the submerged vehicle to create a map of the underwater environment that it is exploring and to track its own position.



(a) (b)

Figure 1.1: The Seabed AUV at the ACFR University of Sydney, Australia. Originally built by WHOI.

Applications such as underground mine mapping are another application for SLAM. Due to the low permeability of the ground to radio waves, which are often used by beacon based localization systems such as GPS, absolute position measurements can be difficult. The Groundhog, shown in Figure 1.2, built by Carnegie Mellon, is an example of a vehicle that is designed to traverse mining tunnels and generate mapping information. Equipped with various sensors, including a 3D laser scanner, this vehicle is able to generate full 3D maps of underground tunnel networks using the SLAM process [30].

Figure 1.2: The Groundhog UGV from Carnegie Mellon University is designed for mapping underground mining tunnels.

Urban environments are another example of an environment where SLAM is useful. Although GPS coverage is often available, the quality may be poor due to reflections from local structures, or reception may be intermittent as the vehicle negotiates local obstacles which block its view of the sky. The dataset used for experiments in this thesis is provided by the Australian Centre for Field Research (ACFR) at the University of Sydney and was collected by the vehicle shown in Figure 1.3 at Victoria Park in Sydney, Australia. As the vehicle travels near trees, satellite reception is blocked and GPS position measurements become unavailable.

Though operation of such vehicles can be done remotely under the control of a human operator, the intelligence to perform the required tasks is shifting from remote to on-board computers to increase productivity and decrease reliance on slow or unreliable communication channels. This requires systems to operate with little or no input from a human controller. In order for these vehicles to perform the required tasks, they must be able to navigate safely and successfully within their environment autonomously.

Navigation is the key component of this problem and requires the generation of maps and the localization of an agent within that map. This need for efficient and accurate map generation has been the focus for significant amounts of research, since this is often

Figure 1.3: The UTE vehicle at the ACFR used to collect the Victoria Park dataset

required when operating in unknown environments.

1.2.1 Localization

The SLAM problem consists of two tightly coupled problems; localization and mapping. Localization allows a system determine its absolute or relative position in an environment. This is a requirement for the vehicle to control its position and move to other locations.

There are many techniques to assist in localization. If the vehicle has some knowledge of an initial position, it may use solely on-board sensors in a process known as dead reckoning. Dead reckoning is the process of estimating the robot's current position based on a previously estimated position and updating that position based upon known speed, elapsed time, and course [32]. The system tracks its own movement through odometry, such as wheel rotation sensors, or inertial sensors. It also tracks an estimated heading, or measures heading with a compass. By adding up many such measurements, a trajectory estimate can be calculated.

As with any real world measurement, there is some error associated with each reading. The result of adding a series of such measurements together results in a cumulative error

which grows over time. Improving these sensors may decrease the amount of error, but it will still continue to grow unbounded as the vehicle explores new terrain.

Localization can also be determined through a system external to the mobile robot. For example, in many outdoor applications, GPS receivers can be used to localize the vehicle based on estimated ranges to satellites in the Earth's orbit. There are other systems which use local beacons, but follow the same triangulation-type procedure to determine a location in some fixed coordinate frame, such as sonar beacons in underwater applications.

Though such systems can exhibit good performance, and do not suffer from long-term cumulative error, in many applications, total reliance on External Localization (EL) systems is not possible, since observation of the external beacons is not always possible. For example, in applications using GPS, satellite signals may be blocked or affected by buildings, land formations or weather conditions. In contrast, with the correct sensor selection, observation of the environment around a robot is almost always possible.

1.2.2 Mapping

A map allows a system to plan its movement. It permits a system to remember features and data from the world around it and associate this information with landmarks in the surroundings. The dynamic creation of such maps allows vehicles to operate in unknown environments. In some situations, this map may be the desired outcome of the operation, or it may be the key to carrying out a higher level task.

With accurate localization, the creation of a map becomes a trivial task of combining sensor readings together. By combining a series of such reading as the vehicle travels, a complete map can be constructed.

Maps can be constructed in numerous ways. Maps can consist of descriptors of features in the environment (such as trees, rocks or buildings) they can be composed of spatial measurements of the environment (such as laser or sonar scans) or a combination of the two.

In order to guarantee a consistent map, accurate vehicle localization is required. In the absence of EL measurements, a best effort map of the environment may be generated relative to the vehicle position estimates. The heavily interdependent nature of localization and mapping is at the core of the process of SLAM.

1.3 Problem Summary

The SLAM algorithm provides a method of building a map of the environment and localizing the vehicle within that map. A typical instantiation of such a system is based on the Kalman filter [33]. For both the vehicle position and environmental landmarks, it maintains estimates of the location and the associated statistical confidence of those location estimates. This means that it is possible to quantify the estimated error and to understand map inaccuracies.

A fundamental problem with this basic implementation is computational complexity, which grows exponentially with the total number of landmarks observed. Additionally, since a full history of measurements is not maintained, there is no way to make corrections to previous observations when new information becomes available, such as when completing a loop and returning to a previously visited location.

The SLAM system begins with no *a priori* knowledge within its location in its environment and so it operates in a local frame of reference, often times defining its starting point as the origin. Incorporating EL sensors, such as GPS, requires the system to work with measurements from different coordinate frames. Since the relationship between these frames is unknown, the system must estimate the transformation between these frames. Due to the reliability issues associated with EL techniques, these measurements may not be available all the time, and therefore cannot be relied on exclusively for localization.

Much work has been done in the areas of managing computational complexity, since this is a practical problem preventing large scale implementation of these techniques.

The goals of this thesis are to:

1. provide a method to supplement a computationally efficient SLAM implementation with the ability to utilize EL information, when available; and

2. use that information to correct previous trajectory and feature location estimates made when external localization measurements were not available.

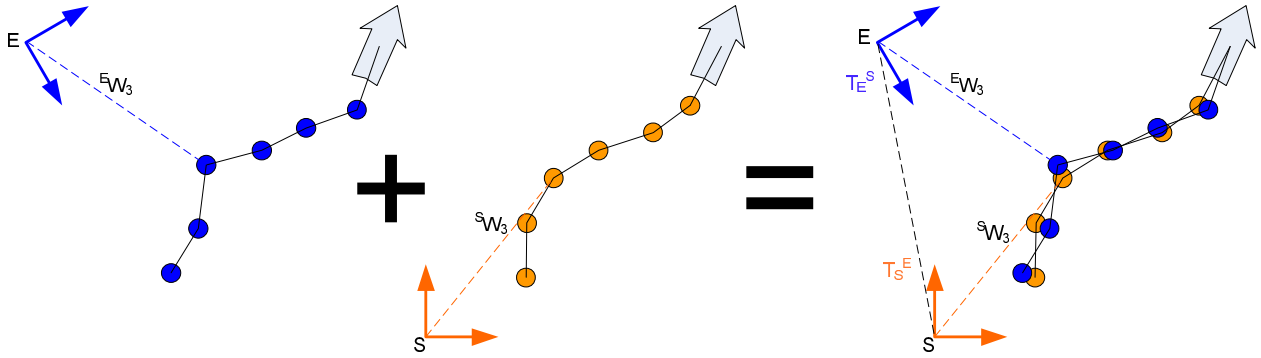


Figure 1.4: Trajectory fitting problem. Blue represents the trajectory as measured in the external frame **E**, and orange as measured in the SLAM frame **S**. ${}^{E}W_3$ represents the third waypoint in the **E** frame, and T_S^E represents the transformation from the **S** to **E** frame.

1.4 Problem Formulation

In order to determine a more effective way to perform the SLAM process, it is necessary to first understand the shortcomings of the existing systems and the reason for those difficulties. Once these are identified, the problem can be accurately defined and the key difficulties to solving each problem can be highlighted and tackled.

As the vehicle moves through the environment its trajectory can be measured by the SLAM system as well as the EL system. As shown in Figure 1.4, if a relationship can be established between the two measured trajectories, the relationship between their coordinate frames can be determined, allowing the use EL in the SLAM system. ${}^{E}W_3$ represents the third waypoint in the **E** frame, and T_S^E represents the transformation from the **S** to **E** frame, and T_E^S its inverse. This can be used to improve SLAM results and eliminate the drift which results from the accumulation of odometry errors.

In order to use the results of a fitting process as described above, a metric of quality must be obtained for the transformation to determine its usability. Also, since one component of the transformation T_S^E is a rotation, as will be explained in more detail later, non-linearities result in the projection of measurement uncertainties. This must be considered when using the transformation.

To combat the problem of exponential growth in computational complexity, systems such as the Constrained Relative Submap Filter (CRSF) SLAM developed by Williams in [35] divide the problem into a series of smaller parts through the concept of multiple submaps, instead of one large environment map. Though the proposed system for using external localization measurements is decoupled from the computational problem addressed through the use of the CRSF algorithm, an additional benefit can be obtained by combining the techniques. An algorithm which uses the EL frame transformation estimates to further refine the CRSF submap locations is presented. This creates a system which can correct the locations of previous submaps, and the features they contain. An advantage not possible with a single map SLAM implementation.

To accomplish the goals presented in the previous section, the proposed solution is:

1. a new method to incorporate external measurements which can be incorporated with the CRSF SLAM algorithm, while preserving its bounded computational complexity; and

2. an update procedure which can improve submap location estimates based on the EL frame transformation estimates of nearby submaps.

1.5 Principal Contributions

This thesis addresses the issues associated with using external localization measurements from unknown coordinate systems within a SLAM system. By utilizing a computationally efficient SLAM technique as its foundation, the solution address the computational complexity issue as well. The major contributions of this thesis are:

- A novel approach to integrating external localization with SLAM by finding a transformation between the SLAM and EL frames

- A novel delayed initialization scheme which incorporates EL only once a good estimate of the transform between frames is available

- A technique to linearize the non-linear projection of transformation errors between coordinate frames

- A technique to utilize EL measurements in CRSF SLAM to update inter-map transformations
- Results of tests conducted on the Victoria Park dataset using the techniques developed herein

1.6 Document Outline

Chapter 2 presents an overview of the SLAM process and various implementations and optimizations that have been made. This section includes a description of the CRSF SLAM filter which a foundation for this research and outlines the motivation for the use of EL within the SLAM framework.

Chapter 3 covers a method for using external localization systems within the SLAM system, specifically detailing a method to incorporate these techniques within CRSF SLAM.

Chapter 4 outlines experimental results from simulations run using this technique. Simulations are conducted on the Victoria Park dataset, which is a common dataset used for testing SLAM algorithms.

Chapter 5 summarizes the contributions of the work presented here and the results of tests in which this technique was applied.

Chapter 2

Simultaneous Localization and Mapping

This chapter gives an outline of the Simultaneous Localization and Mapping (SLAM) process and points out some implementation difficulties with the Extended Kalman Filter (EKF) SLAM algorithm. The benefits of a multi-frame SLAM algorithm, such as Constrained Relative Submap Filter (CRSF) SLAM are presented and the reasons for integrating External Localization (EL) are discussed.

2.1 Introduction

The SLAM algorithm first appeared in a paper by Smith, Self and Cheeseman [33], which built on work by Ayache and Faugeras[2]. Many implementations of the concept of SLAM exist. A fair amount of research has been given to stochastic techniques which perform tightly coupled estimates of vehicle and map feature locations.

A vehicle typically begins with no *a priori* knowledge of its location or surroundings. By discerning features from measurements of its environment, it is able to compute relative observations of nearby landmarks. This allows the system to build a map of the world around it, and simultaneously determine its relative position. As the vehicle continues to move, a more complete map is built and an estimate of its relative location is continuously updated.

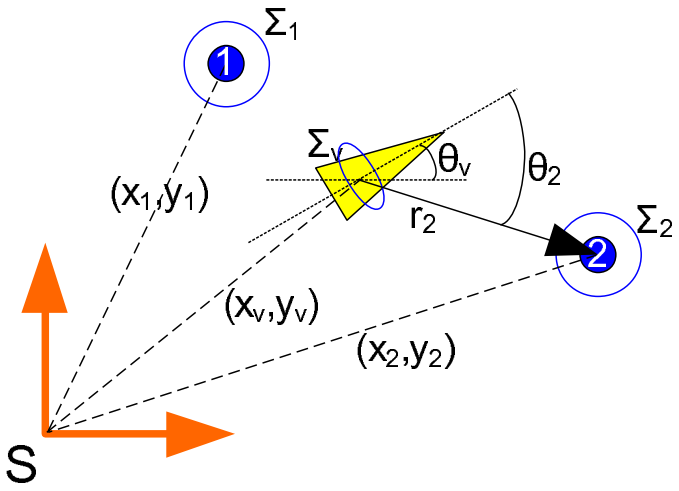


Figure 2.1: A graphical representation of the variables used in EKF SLAM

2.2 EKF SLAM

SLAM was first coined by Leonard and Durrant-Whyte in [22]. Their technique was based on the EKF. The Kalman Filter is an efficient recursive filter that estimates the state of a system from a series of incomplete and noisy observations. The filter uses uncertainty metrics associated with each measurement made to compute an uncertainty of the system state estimate. The EKF is an extension of the Kalman Filter which permits the use of non-linear observation functions by linearization through a first order Taylor series expansion. A very brief description of the EKF SLAM process will given here.

By defining the system state to include the vehicle's position and landmark locations, a map of the landmarks and their associated uncertainties can be computed, making it a useful filter to implement SLAM.

Figure 2.1 shows an example of a 2-dimensional mapping scenario. The vehicle state $\mathbf{x}_v$ consists of three variables, $\mathbf{x}_v = (x_v, y_v, \theta_v)^T$. Shown are two landmarks $\mathbf{x}_1$ and $\mathbf{x}_2$, with covariance ellipses Σ_1 and Σ_2. Each landmark has its estimated location specified by $\mathbf{x}_n = (x_n, y_n)$. These measurements are all with respect to the SLAM frame $\mathbf{S}$. As the vehicle travels through its environment, it gathers range and bearing estimates to the

features around it. One such measurement, $\mathbf{z}_2$, is shown as $\mathbf{z}_2 = (r_2, \theta_2)$. Measurements are taken with respect to the vehicle's frame.

The full system state, $\mathbf{x}(k)$, at sample time k is given by the concatenation of all vehicle and map states:

$$\mathbf{x}(k) = \begin{bmatrix} \mathbf{x}_v(k) \\ \mathbf{x}_1(k) \\ \vdots \\ \mathbf{x}_n(k) \end{bmatrix} \tag{2.1}$$

or more concisely:

$$\mathbf{x}(k) = \begin{bmatrix} \mathbf{x}_v(k) \\ \mathbf{x}_m(k) \end{bmatrix} \tag{2.2}$$

where $\mathbf{x}_m(k)$ is the subset of $\mathbf{x}(k)$ containing all the map states.

In addition to the state estimates, uncertainty estimates are maintained in the form of a covariance matrix, Σ_x. These are shown as blue ellipses in Figure 2.1. The covariance matrix is composed as:

$$\Sigma_{\mathbf{x}} = \begin{bmatrix} \Sigma_v & \Sigma_{v,1} & \cdots & \Sigma_{v,n} \\ \Sigma_{v,1} & \Sigma_1 & \cdots & \Sigma_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{v,n} & \Sigma_{1,n} & \cdots & \Sigma_n \end{bmatrix} \tag{2.3}$$

where Σ_v is the vehicle covariance, Σ_n is the covariance of the n^{th} landmark state and $\Sigma_{i,j}$ is the covariance of i^{th} feature state with repsect to the j^{th} feature.

It is assumed that the vehicle travels according to a discretized first order non-linear vector differential equation model:

$$\mathbf{x}_v(k) = \mathbf{f}(\mathbf{x}_v(k-1), \mathbf{u}(k)) + \mathbf{v}_v(k) \tag{2.4}$$

where $\mathbf{u}(t)$ is a known control input at time t and $\mathbf{v}_v(k)$ is a random vector describing the dynamic noise associated with state propagation. In order to fit within the EKF framework, this noise is modeled with a multivariate zero-mean Gaussian distribution, with a covariance of $Q(k)$. Landmarks in the environment are assumed to be stationary.

Observations of features in the environment may require a transformation to be used by the filter, and the measurements taken are also subject to some errors. The discretized

observation process is modeled as:

$$\mathbf{z}_n(k) = \mathbf{h}(\mathbf{x}(k), n) + \mathbf{w}(k) \tag{2.5}$$

where $\mathbf{z}_n(k)$ is the observation of feature n made at time k, $\mathbf{h}(.)$ is a model of the observation transformation as a function of the system state and $\mathbf{w}(k)$ is a zero-mean Gaussian random vector describing the measurement noise, with a covariance of $R(k)$.

For the example given in Figure 2.1, the a range-bearing measurement, given in Equation 2.5 of feature n at time k is given by:

$$\mathbf{z}_n(k) = \begin{bmatrix} \sqrt{(x_v(k) - x_n(k))^2 + (y_v(k) - y_n(k))^2} \\ \arctan(\frac{(y_v(k) - y_n(k))}{(x_v(k) - x_n(k))}) - \theta_v(k) \end{bmatrix} + \begin{bmatrix} w_r(k) \\ w_\theta(k) \end{bmatrix} \tag{2.6}$$

where $w_r(k)$ is the range measurement error at time k and $w_\theta(k)$ is the bearing error at time k.

EKF SLAM calculates an estimate, $\hat{\mathbf{x}}(k)$ of the actual state $\mathbf{x}(k)$. This process consists of two main steps: a prediction step and an update step. The prediction step runs each time a control input is available. The estimates advances according to: [22]

$$\hat{\mathbf{x}}(k|k-1) = f(\mathbf{x}(k|k-1), \mathbf{u}(k)) \tag{2.7}$$

$$\Sigma(k|k-1) = \mathbf{F}(k)\Sigma(k-1|k-1)\mathbf{F}(k)^T + Q(k) \tag{2.8}$$

$$\tag{2.9}$$

where the notation $(k|k-1)$ indicates a result at time k based on information from time $k-1$, and $\mathbf{F}$ is the Jacobian of the time propagation function $f(.)$ with respect to the system state $\mathbf{x}$. The Jacobian of $f(.)$ is obtained by taking the partial derivative of the function $f(.)$ with respect to each of the variables of the state.

The update step is conducted for each observation of landmarks available. This is done with the following equations:

$$y(k) = z(k) - h(\hat{\mathbf{x}}(k|k-1)) \tag{2.10}$$

$$S(k) = \mathbf{H}(k)\Sigma(k|k-1)\mathbf{H}(k)^T + \mathbf{R}(k) \tag{2.11}$$

$$K(k) = \Sigma(k|k-1)\mathbf{H}(k)^T S(k)^{-1} \tag{2.12}$$

$$\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + K(k)y(k) \tag{2.13}$$

$$\Sigma(k|k) = (I - K(k)\mathbf{H}(k))\Sigma(k|k-1) \tag{2.14}$$

where $y(k)$ is known as the *innovation*, $K(k)$ is known as the *Kalman gain* and I is the identity matrix. $\mathbf{H}(k)$ is the Jacobian of the observation function $h(.)$ with respect to the system state $\mathbf{x}$.

This provides the basic structure for the EKF SLAM system. In addition to the basic filter, a procedure to associate measurements with their corresponding landmarks is required. Details of the implementation used for this research are given in Section 4.2.2.1.

2.3 EKF SLAM Shortcomings

There are numerous shortcomings with a basic EKF SLAM implementation. These include:

- exponential growth in Kalman filter complexity as the map grows
- the inability to use EL measurements directly
- the inability to correct the map for loop closure

Loop closure is the term used to describe a scenario in SLAM which occurs when a vehicle returns to a location where it's been before. Through proper feature association, this allows the filter to correct the vehicle's location, but it cannot update existing map feature locations to correct for vehicle position error.

A more ideal SLAM implementation would have the ability to address the issues above. The goals of the method being developed include:

1. bounded computational complexity
2. the ability to deal with large, complex environments
3. efficient loop closing
4. ability to utilize with EL when available

As a field of active research, several SLAM methods have been proposed which accomplish some of the goals stated above. For example, the need to reduce computational has be recognized and techniques to optimize EKF SLAM implementations have been proposed to reduce the computational complexity [3, 14, 13, 24, 11].

Others have developed are variations to the basic filter including particle filter approaches such as FastSLAM, which operate with multiple vehicle position hypotheses [28]. The approach studied here is the CRSF algorithm developed by Williams [35]. It follows the basic EKF SLAM structure, but bounds the computation complexity by breaking the problem into a series of bounded size submaps. Both [11] and [35] present similar approaches, however the work in [11] focuses on the loop closing issue in the multi-map paradigm.

To incorporate EL systems, some approaches work directly in the external coordinate frame. For example, in [21], the SLAM filter uses *a priori* knowledge of the vehicle's starting point to define the SLAM coordinates. This avoids the problem of determining a transform between the SLAM and external frame, since they are the same frame, but limits the ability of the filter to operate without external localization measurements. Overall, there seems to be little research into using EL systems efficiently within the SLAM paradigm.

2.4 SLAM with multiple coordinate frames

The chicken-or-egg relationship between localization and mapping is a consequence of how errors in the robot's sensor readings are corrupted by error in the robot's motion. As the robot moves, its pose estimate uncertainty grows because of motion noise. Motion noise is a total accumulation of errors including, for example, sensor noise, wheel slip, and unmodeled environmental disturbances. The perceived locations of objects in the world are, in turn, affected by both measurement noise and the error in the estimated pose of the robot. However, unlike measurement noise, error in the robot's pose will have a systematic effect on the error in the map. More directly, error in the robot's path results in errors in the map. As a result, the true map cannot be estimated without also estimating the true path of the robot.

Since the robot's position uncertainty grows as the map grows, an approach to limiting the vehicle's uncertainty is to create a series of submaps. Whenever the current submap grows too large or the vehicle leaves the map's boundaries, a new map is created. Its origin is placed at the current location of the vehicle, with its x-axis aligned to the vehicle's x-axis. The transformation which relates the previous map and the new map, is the location and

orientation of the vehicle from the previous map, which is $\mathbf{x}_v$ in the system state.

Though not known accurately in an absolute sense, by moving to a new submap, the vehicle's position in the new submap is known with no uncertainty. This allows a more accurate local submap to be built. By maintaining the relative transformations between these submaps, a group of submaps can take the place of one larger one.

2.4.1 CRSF SLAM

The CRSF SLAM algorithm was first presented by Williams as an extension to his Constrained Local Submap Filter (CLSF) algorithm [35]. The CRSF algorithm works based on the principle described above, in which SLAM submaps are created to represent subsets of the overall map.

Figure 2.2 graphically depicts an example of several submaps, each representing portions of the vehicle's environment. Within each of these submaps, the EKF SLAM algorithm runs to track the vehicle's position and map feature locations.

In order to relate each of the submaps, transformations between these maps are maintained. For example, in Figure 2.2, the location of the original frame F_{L2} in frame F_{L1} is given by ${}^{L1}x_{L2}$. This is the relative translation between the two frames. This transformation is estimated at first using the vehicle's location in F_{L1} when F_{L2} is created. As can be seen from the Figure, some features overlap the two maps. This is shown as multiple feature ellipses in approximately the same location, resulting from measurements of the same feature in each frame. Since these map features represent the same physical features, a constraint satisfaction approach is applied to correct the map transformation to minimize the distance between the locations of the features between the two maps [35].

Figure 2.3 shows an example of the projection of a feature from the F_L frame into the F_G frame and the subsequent constraint satisfaction. The feature constraint satisfaction is based on the notion that the observations are of the same feature so the following must hold:

$$ {}^{G}x_L + {}^{L}x_i = {}^{G}x_i \tag{2.15}$$

This equation is formulated into a Kalman filter update to apply these constraints, the details of which are given in [35]. The algorithm works by performing a weighted adjustment to the feature position in each frame as well as the transformation between each

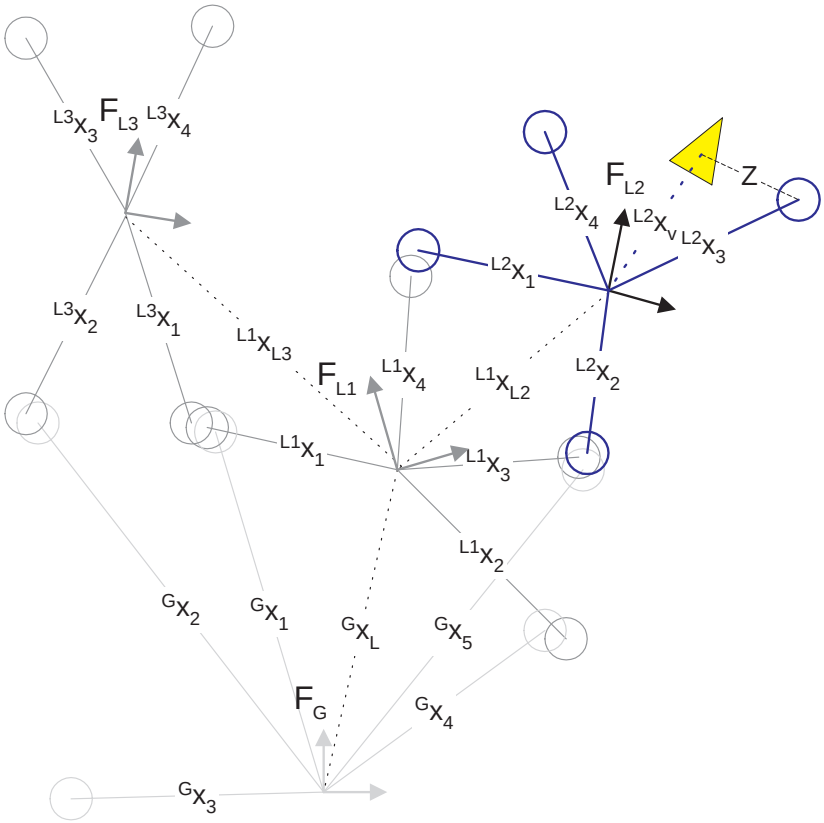


Figure 2.2: Example CRSF map. Four frames are shown, a global frame F_G, which could represent an EL frame, and three local frames F_{L1}, F_{L2} and F_{L3}. A series of feature location estimates are shown, some of which overlap multiple submaps. They are represented by circles, with their centres at the esimated position. Since each frame is indepdent of all others, they each use their own indices to refer to landmarks. No vehicle trajectory is shown. Source [35].

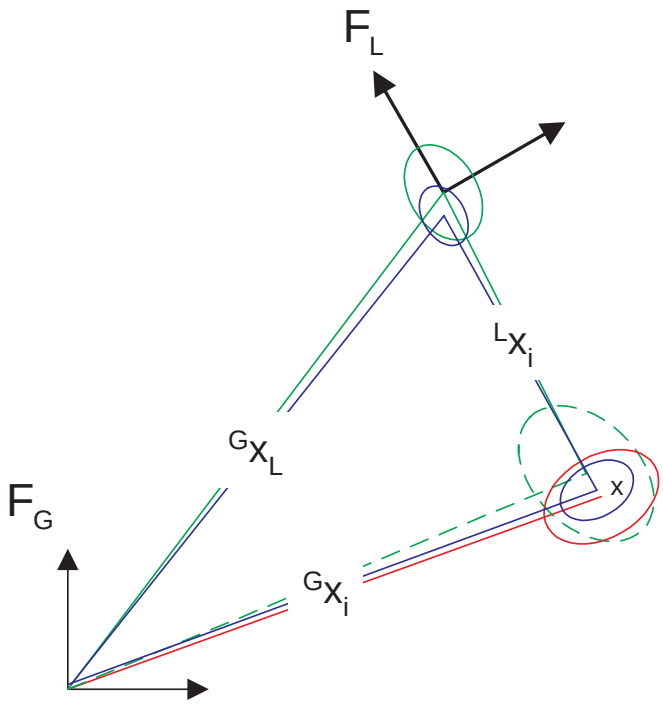


Figure 2.3: CRSF constraints. The dashed green line shows the estimate of feature i generated through the frame F_L while the estimate in the F_G frame is shown by the solid red line labeled ${}^{G}x_i$. After constraints are applied, the updated frame transformation estimate and the estimate of the feature location are shown in blue. Source [35].

frame based on the constraint specified in Equation 2.15. As in the EKF update step, the inverse of the uncertainty of the feature estimates and transformation estimate are used as weighting factors.

Using shared *feature constraints* allows the CRSF system to correct the relative transformations between frames as well as improve the feature location estimates in each map. This can be seen in Figure 2.3 through the reduced uncertainty for the constrained estimates as shown in blue.

2.4.1.1 Loop Closure

Though the application of these inter-map feature constraints is independent of the map transformation, no technique is given in [35] to perform loop closure and propagate these corrections along a sequence of related submaps. A goal of this research is to provide a method for doing this.

2.5 External Localization

The use of EL is not a capability of the original CRSF algorithm. Adding EL functionality to CRSF SLAM is a goal of this research.

Figure 2.4 gives an example of a SLAM scenario. In this figure, we see a local coordinate system defined, with its origin at the fence post shown. The system is able to accurately measure the location of features with respect to this origin, and if the system can compute the relative transformation between the local map coordinate system and a global coordinate system defined by the EL system, feature and trajectory information could be presented in either reference frame.

There are several difficulties to overcome when trying to incorporate EL measurements into a relative coordinate frame SLAM system, such as CRSF SLAM. First, once a transformation between frames is estimated, determining the "quality" of the transformation is not an obvious process. One may assume that the residuals from the fit would be used, guessing that the magnitude of the errors represents the uncertainty of the state, but it will be shown that the covariance of the transformation is independent of the residuals.

Figure 2.4: A graphical representation of local and global coordinates in multiple coordinate frame SLAM. Local feature locations are shown with blue lines, and feature observations from the vehicle are shown as red lines. Source [16].

Next, in addition to transforming the mean of the measurements between frames, the associated covariances must be transformed as well. The observation function, which converts EL measurements into the local SLAM frame, is non-linear and, therefore, results in a non-Gaussian Probability Distribution Function (PDF) for the transformed measurement. This must be linearized before it can be used in the SLAM filter.

These challenges will be addressed by the procedure presented in the following chapter.

Chapter 3

Multi-frame Measurement Fusion

The core problem to utilizing External Localization (EL) within the Constrained Relative Submap Filter (CRSF) Simultaneous Localization and Mapping (SLAM) framework revolves around the ability of the system to estimate the transformation between the current local frame and the EL reference frame. In order to estimate this relationship, EL measurements must be obtained at the same time as SLAM vehicle position estimates are generated. By correlating these two sets of data, an estimate of the transformation between these two sets of data can be determined.

Once a transformation is determined, this technique can be used as an extension to the CRSF algorithm to allow EL to be used within local frames, as a constraint between relative frames and during loop closing. Once a transformation is available, it can aid the system by providing a means to determine the vehicle's position in the EL frame during periods of EL drop-out.

3.1 Process Flow

This chapter proposes a system which operates on two sets of input data, local SLAM vehicle position estimates and external localization position estimates. By using the uncertainty data which accompanies each input dataset, a transformation can be computed based on a weighted fit of the correlated position data. Figure 3.1 outlines the process that will be presented in this chapter.

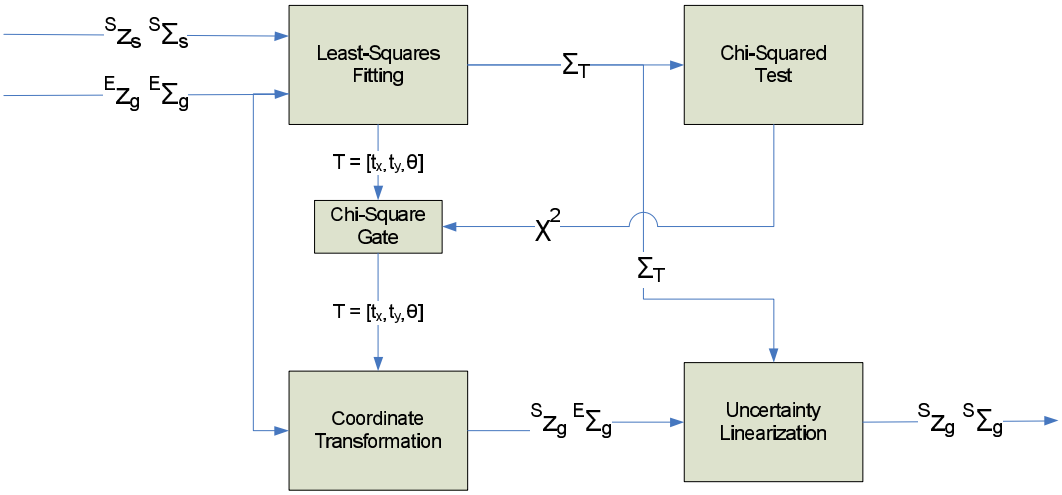


Figure 3.1: Coordinate Fitting Flow

The process works by first collecting localization data from both the internal and external reference frames. The measurement data in the SLAM frame **S** is denoted as ${}^{S}z_s$, with the corresponding covariance matrix of ${}^{S}\Sigma_s$. The subscript s indicates these measurements from from the SLAM process. EL measurements, in the **E** frame are denoted by ${}^{E}z_g$ and ${}^{E}\Sigma_g$, where the subscript g is used to denote the Global Positioning System (GPS) as the source of these measurements. These measurements must be collected in a synchronized fashion to ensure correlation of the sets. The process which fits these maps together, to come up with the relative transformation **T**, is outlined in Section 3.2.

Once a fit has been determined, the uncertainty Σ_T of the transformation between the data sets is determined with the procedure presented in Section 3.3. A chi-squared (χ^2) test, as described in Section 3.3.1 is used to validate the results of the fitting process. This validation is implemented within a gating function that allows the fusion of the external measurements with local measurements only when the chi squared test is passed. Section 3.4 discusses the reasons for delaying the use of this transformation until it reaches a predetermined quality.

Once a suitable transformation has been determined, subsequent localization measurements from the external frame can be transformed to local coordinate frame measurements

so they can be incorporated for localization in SLAM. In order to use the normal SLAM filter update procedure, the uncertainty of these measurements must be expressed as the variance of a Gaussian distribution. Section 3.5 provides a technique for linearizing the uncertainty to incorporate measurements with a SLAM filter based on the Extended Kalman Filter (EKF).

When the above procedure is used within CRSF SLAM, it is likely that not all submaps will obtain suitable transformations between that submap's local frame and the EL system frame. This is especially true in situations with spurious GPS coverage, such as urban canyons. Section 3.6 presents a technique to refine the inter-map transformation estimates for those maps in the system with undetermined EL transformations when surrounding submaps have valid external localization transformations.

Lastly, Section 3.7 gives some extensions which can be used in a system to incorporate EL measurements, before and after a transformation has been fully determined, through an appropriate data reuse procedure.

3.2 Map Fitting

The map fitting process takes two synchronized sets of vehicle trajectory measurements and attempts to find a transformation which relates those two sets of data. Figure 3.2 depicts this graphically. The trajectory ${}^{E}\mathbf{W}$ depicted in blue is as measured by the EL system in the external frame $\mathbf{E}$. This consists of n way points such that ${}^{E}\mathbf{W} = [{}^{E}W_1, {}^{E}W_2, \ldots, {}^{E}W_n]$. Similarly, the trajectory in the SLAM frame $\mathbf{S}$ is denoted by ${}^{S}\mathbf{W}$.

In implementation, EL data could be data gathered by various EL systems including GPS receivers, radio beacon triangulation systems or a sonar beacon network [29]. This frame is externally referenced to some known, fixed location. In the case of GPS, this is the Earth-centred Earth-fixed (ECEF) frame shown in Figure 3.3 [12]. In Figure 3.2, shown in orange, is the data in the SLAM $\mathbf{S}$ frame. This frame is relative to the starting location of the vehicle's map. This reference point will change with the initialization of the vehicle and in the case of CRSF SLAM with the creation of each new submap.

In order to determine a transformation, $\mathbf{T}^{\mathbf{E}}_{\mathbf{S}} = (t_x, t_y, \theta)$, which relates the external and SLAM frames and its inverse $\mathbf{T}^{\mathbf{S}}_{\mathbf{E}}$, a correlation of the data must be performed. This can be

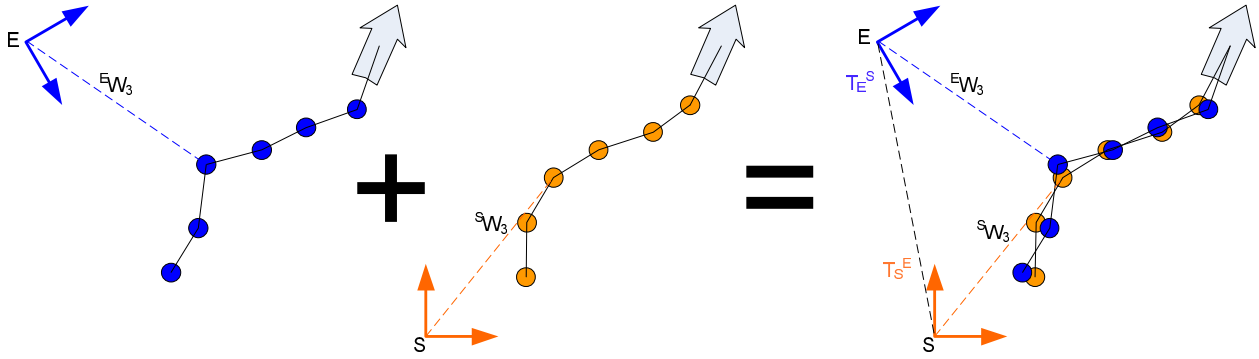


Figure 3.2: Trajectory measurements from two frames. Blue represents the trajectory ${}^{E}W = ({}^{E}W_1, {}^{E}W_2, \ldots, {}^{E}W_n$ as measured in the external frame, and orange ${}^{S}W$ as measured in the SLAM frame **S**

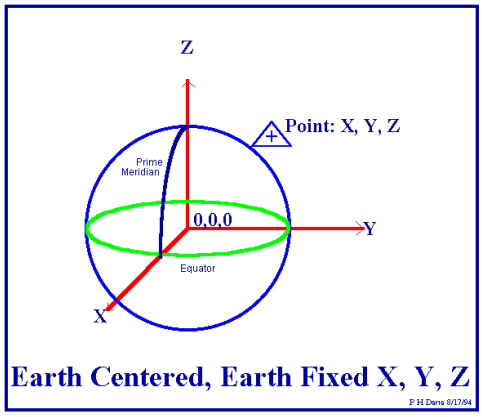


Figure 3.3: The ECEF coordinate frame [10]

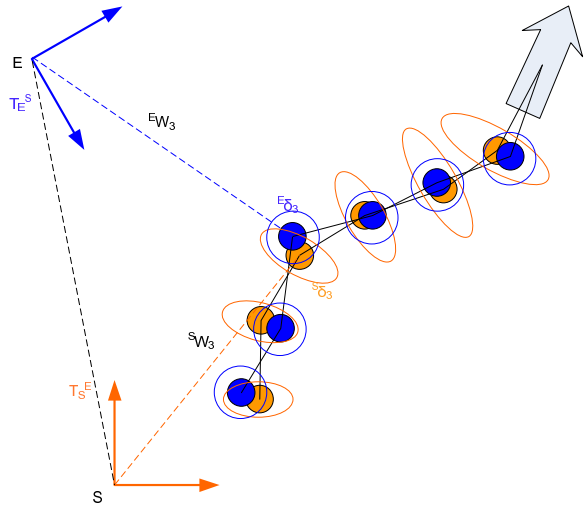


Figure 3.4: Uncertainty measurements in map fitting

done in the form of a minimization which attempts to minimize the distance between the two trajectories. This is the first major step in the map fitting process given in Figure 3.1.

Since both SLAM and GPS measurements can be gathered on the vehicle simultaneously, the matching of the two sets of data points is trivial. An example trajectory way point W_3 is shown in Figure 3.2. Due to measurement errors in the SLAM process and GPS inaccuracies, the measured trajectories ${}^SW$, in the SLAM frame, and ${}^EW$, in the external frame, will differ.

The fitting process can be designed to minimize the distance between the two trajectories in a least-squares sense. An additional variable which should be considered in this process is uncertainty. Measurement error in both the SLAM process and the external measurement system can be estimated. In the SLAM process, the calculation of uncertainty is core to the operation of the Kalman filter, and is available in the covariance matrix as the vehicle position uncertainty. For EL, measurement error statistics will vary depending on the EL system used. For some systems error may be a function of distance from stationary beacons. For GPS, error is a function of satellite positions, atmospheric effects and many other factors [12]. In either case, it is possible for both systems to provide a metric of measurement uncertainty.

Figure 3.4 shows the same scenario as Figure 3.2. However, covariance estimates are now shown. The ellipses drawn represent one standard deviation of uncertainty for the measurements. As the vehicle travels further from the origin of the SLAM map, the error estimate grows. The SLAM measurements grow in uncertainty because as the vehicle moves, errors compound. The GPS measurements, however, are not a function of the distance that the vehicle travels, and remain constant and independent of location. In a real-world implementation GPS errors may not remain constant due to measurement conditions, but they remain independent of the SLAM system uncertainty. With this in mind, the objective function of this minimization becomes the the sum of the distances between the external and local SLAM measurements, weighted by the inverse of their respective uncertainties.

More formally, n SLAM trajectory measurements are collected in the SLAM **S** frame ${}^{\mathbf{S}}\mathbf{z_s}(j) = ({}^{S}z_{s,x}(j), {}^{S}z_{s,y}(j))$, where $j = 1 \ldots n$, with uncertainties of ${}^{S}\Sigma_s(j)$. In a synchronized fashion, n GPS measurements are gathered in the external **E** frame and denoted by ${}^{\mathbf{E}}\mathbf{z_g}(j)$ with uncertainties of ${}^{E}\Sigma_g(j)$. Since the measurements are obtained together, ${}^{S}z_s(j)$ and ${}^{E}z_g(j)$ are measurements of the vehicle at the same location. There exists an unknown transformation $\mathbf{T}^{\mathbf{E}}_{\mathbf{S}} = (t_x, t_y, \theta)$ which relates the two coordinate frames. This is shown graphically in Figure 3.5.

Once the parameters of this transformation are found, the equation which expresses the SLAM measurements in the external frame is:

$$ {}^{E}z_s(j) = R(\theta) {}^{S}z_s(j) + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{3.1} $$

where $R(\theta)$ is given by the 2-d rotation matrix:

$$ R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{3.2} $$

3.2.1 The Horn Algorithm

If this were a linear problem with a quadratic cost, there would be an explicit solution. The fitting problem here is a highly nonlinear problem that needs a numerical solution. As with any minimization process, an initial guess at the system parameters is required.

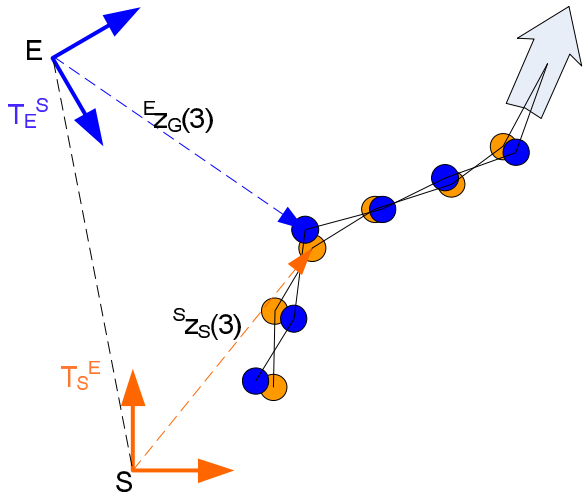


Figure 3.5: Way point measurements in multiple coordinate systems

This initial guess must be close to the true solution in order to ensure rapid convergence of the system and to avoid local minima.

For this step of the procedure, the Horn algorithm was selected [17]. This procedure is a closed form solution to determine the transformation between two coordinate systems given a corresponding series of points from each frame. It does not provide a final solution to the fitting problem, since it finds only a mapping between the centroids of the data, and does not take into account uncertainty. This algorithm is ideal because it provides an $O(n)$ solution to find initial conditions, where n is the number of trajectory point pairs used.

There are two main steps to this algorithm. The first finds the centroids of the two data sets and determines the offset, the next determines a rotation about that centroid which best aligns the data. This is depicted in Figure 3.6.

The two data sets are shown in orange and blue. The observations are subject to some error, so the relative location of each of the points to its origin are not identical between frames. The geometric centroid of each of the datasets is depicted with an X.

Given a set of n points, the centroid is calculated as the first moment of the data

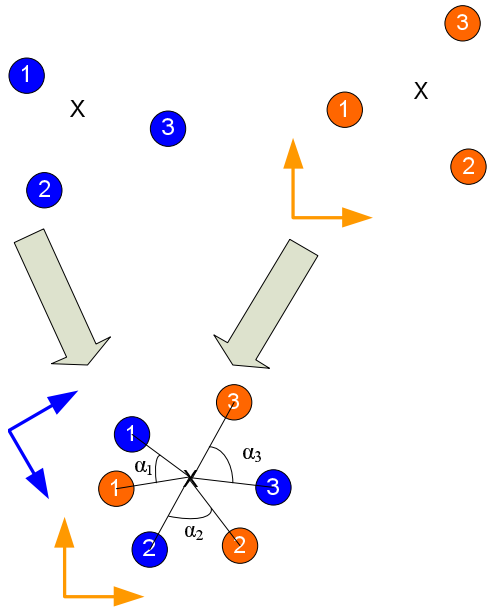


Figure 3.6: The Horn Algorithm

according to:

$$\bar{r} = \frac{1}{n}\sum_{j=1}^{n}\vec{r}_j \tag{3.3}$$

The translation $\vec{t}_S^E$ which relates the external and SLAM frames is, therefore:

$$\vec{t}_S^E = {}^S\bar{r}_s - {}^E\bar{r}_g \tag{3.4}$$

where ${}^S\bar{r}_s$ and ${}^E\bar{r}_g$ are the centroids. To find the rotation, the next step examines the angles between corresponding points, by first moving the centroid of each cluster to the origin and then examining the radial angle between sets of transformed points and the origin. Let us denote:

$${}^E\vec{r'}_{g,j} = {}^E\vec{r}_{g,j} - {}^E\bar{r}_g \tag{3.5}$$

and

$${}^S\vec{r'}_{s,j} = {}^S\vec{r}_{s,j} - {}^S\bar{r}_s \tag{3.6}$$

Selecting one of the frames as a reference, say the **S** frame, we compute the relative angles between sets of transformed points. In Figure 3.6 these are denoted as α_n. The best-fit rotation between frames is then computed as:

$$\theta = \frac{1}{n}\sum_{j=1}^{n}\alpha_j \tag{3.7}$$

Though the result of this algorithm is a transformation $\mathbf{T} = [\vec{t}, \theta]^T$, this fit does not account for the uncertainty in the data, since the covariance data was not used in this process. It does, however, provide a reasonable initialization value for the transformation parameters used in the minimization presented in the next section.

3.2.2 Levenberg-Marquardt Minimization

As was mentioned previously, the measurement uncertainty is used as a weighting in the minimization process. The Levenberg-Marquardt minimization is selected for this role, as a general non-linear downhill minimization algorithm [26]. The specific notation presented here is derived from Kalman Filtering theory [5].

This algorithm operates by attempting to minimize the Mahalanobis distance between an actual measurement and an expected measurement [25]. The measurement function $z(j)$ has the general form of:

$$z(j) = h(j; \mathbf{x}) + w(j) \tag{3.8}$$

where $h(j; \mathbf{x})$ is the observation function for the j^{th} point of the system with the measurement parameters $\mathbf{x}$. This measurement includes $w(j)$, a Gaussian random variable, with zero-mean and a covariance of $N(j)$.

For this minimization, we have two sets of measurements which must be combined in a way that fits the structure of this minimization. In order to do this, we arbitrarily choose to operate in **E** frame. We use ${}^{E}z_g(j)$ in place of $z(j)$ in Equation 3.10 and use ${}^{E}z_s(j)$ as defined in Equation 3.9 as $h(.)$:

$$h(j) = R(\theta)^{S}z_s(j) + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{3.9}$$

This minimizes the distances between the two trajectories, by evaluating the distance between the GPS measurements taken directly in the **E** frame and the SLAM measurements transformed into the **E** frame, based on the current transformation estimate.

Since the trajectories are in different coordinate frames, we use a variation of the Mahalanobis distance. The development of this variation is presented in Appendix A.

Next, we introduce $\hat{\mathbf{x}}$, which is an estimate of the actual system parameters. The maximum likelihood solution minimizes:

$$\chi^2(\hat{\mathbf{x}}) = \sum_{j=1}^{k} (z(j) - h(j;\hat{\mathbf{x}}))^T N(j)^{-1} (z(j) - h(j;\hat{\mathbf{x}})) \tag{3.10}$$

The similarity between the Mahalanobis distance given in Equation A.2 and Equation 3.10 should be noted. The noise of this overall measurement process, $N(j)$, is required when calculating the distance between measurements. This is analogous to the covariance in the Mahalanobis distance. Since two sets of measurements are involved in this calculation, the covariances of the two sets will be combined. It is assumed that the two sets of measurements are uncorrelated and independent, therefore, their noises are assumed to be additive as discussed in Appendix A. However, before they can be summed, the uncertainty estimate from the SLAM frame must be transformed into the **E** frame. The rotation matrix $R(\theta)$ is again used for this step. The resulting $N(j)$ is given by:

$$N(j) = R(\theta)^S \Sigma_s(j) R(\theta)^T + {}^E\Sigma_g(j) \tag{3.11}$$

3.3 Fit Uncertainty

The uncertainty of the best fit solution from the minimization above is independent of the residual errors of the parameter fitting. It is only dependent on the covariances of the measurements used and is evaluated directly from the minimization [6]. Here, the covariance of the transformation $\mathbf{T}_S^E$ is given by: [26]

$$\Sigma_T = A^{-1} \tag{3.12}$$

where:

$$A = \sum_{j=1}^{k} H(j;\hat{\mathbf{x}})^T N(j)^{-1} H(j;\hat{\mathbf{x}}) \tag{3.13}$$

where $H(j;\hat{\mathbf{x}})$ is the Jacobian of $h(j;\hat{\mathbf{x}})$ with respect to the minimization parameters, evaluated at $\hat{x}$. The Jacobian of Equation 3.9 is

$$H(j) = \frac{\partial h(j)}{\partial T_S^E} \tag{3.14}$$

$$H(j) = \begin{bmatrix} \frac{\partial h(j)}{\partial t_x} & \frac{\partial h(j)}{\partial t_y} & \frac{\partial h(j)}{\partial \theta} \end{bmatrix} \tag{3.15}$$

$$H(j) = \left[\begin{array}{cc|c} 1 & 0 & \\ 0 & 1 & R'(\theta)^S z_s(j) \end{array}\right] \tag{3.16}$$

where $R'(\theta)$ is the derivative of $R(\theta)$ with respect to θ:

$$R'(\theta) = \frac{\partial R}{\partial \theta} = \begin{bmatrix} \sin(\theta) & \cos(\theta) \\ -\cos(\theta) & \sin(\theta) \end{bmatrix} \tag{3.17}$$

It can be seen that the Jacobian is dependent upon the measurement ${}^S z_s$. This results from the rotation at the origin of the **S** frame. This sensitivity occurs since the further a measurement is from the origin, the more effect a change of rotation has on the location of that measurement in the new frame.

3.3.1 Chi Squared Test

The uncertainty resulting from the calculations above only holds true if there is sufficient statistical significance in the data available. To test this conditions we use the chi-squared test, which is a statistical hypothesis test [9].

By examining the ratio of magnitude of the normalized residuals to the degrees of freedom of the system, the above condition is tested [26]:

$$\beta = \frac{\chi^2}{(n-p)} \tag{3.18}$$

where n is number of data points used from each trajectory and p is the number of parameters in the system state. The system state here is the the transformation being sought,

which are the variables of $\mathbf{T}_{\mathbf{S}}^{\mathbf{E}}$ for this minimization, so $p = 3$. The fit is statistically significant, and therefore meaningful and acceptable for use, if $\beta << 1$ [9]. If this is not the case, the transform estimate uncertainty estimate is invalid, and cannot be used.

3.4 Delayed Initialization

The EKF SLAM filter does not keep measurement history. It only tracks the current system state. It does not have a mechanism to undo and correct previous measurements and updates made to the SLAM map. This means that once a transformation is being used to incorporate EL measurements with the SLAM filter, it cannot be changed, since measurements taken with previous versions of the transformation cannot be updated. Performing corrections on previously measurements would advantageous in situations such as loop closure. It would allow the system to correct landmark position estimates in the map with errors resulting from incorrect vehicle position estimates, which were later corrected by loop closure.

Instead, the approach taken is to delay the initialization of the transformation until its certainty reaches a desired "quality". This means that the SLAM algorithm runs as normal until a transformation of the desired uncertainty is obtained, at which point measurements from the EL system can be incorporated into the SLAM update step to improve vehicle localization. This technique is adapted from [23] and [4].

Delaying the initialization of the transform allows the system to minimize, as much as possible, the non-linear effects of the uncertainty of the transform variable θ as well as the translational (t_x, t_y) components. The measure of quality, in this case, are maximum uncertainties for the 3 parameters of the transformation. For example, setting a criteria of $3\sigma_\theta < 1^\circ$ will assure, with 99.7% probability, that rotational uncertainty is less than 1 degree.

By examining the residuals from the minimization and the degrees of freedom of the system using the chi-squared test and the uncertainty estimate of the system, we form a usage condition.

A design trade-off exists in the selection of the gating condition. The more strict the maximum acceptance requirement, the longer it will take to collect sufficient data for

initialization of the transformation. This delays the use of EL for aiding map generation.

3.5 Linearization of Uncertainty for EKF

Once a satisfactory transformation is determined, the goal is to be able to use this transformation to convert the EL measurements into a form directly usable in the EKF SLAM process. This means that the measurements must be transformed into the SLAM frame **S**. The details of an implementation are presented in this section.

As was seen in Equation 3.16, the Jacobian of $h(.)$ is dependent upon the distance of the trajectory point from the origin of the SLAM frame. This sensitivity to the rotation θ in the transformation introduces a nonlinearity. This can be seen in Figure 3.7. The GPS measurement ${}^{E}z_g(3)$ is transformed through T_E^S into the **S** frame. The transformed measurement is shown as ${}^{S}z_g(3)$ in black.

The uncertainty must be also transformed into the **S** frame. This transformation is not as straightforward. The Kalman Filter requires that measurements be in the form of multidimensional random variables with Gaussian distributions. Since the filter is operating in the **S** frame, GPS measurements must be converted using the T_E^S. This transformation, however, has some uncertainty, Σ_T.

In Figure 3.7, the uncertainty Σ_T has been shown as two components $\Sigma_{x,y}$ and Σ_θ which represent the translational and rotational components. The translational component is independent of GPS measurement noise, so can be added directly to the measurement noise. Combining the translational and rotational components creates a non-Gaussian Probability Distribution Function (PDF) in the **S** frame, shown as the black locus. In order to be used by the EKF SLAM filter, this must be converted to a multi-variate Gaussian respresentation.

A solution to this problem is to over-estimate the uncertainty by fitting an ellipse around the locus. This ellipse can be expressed as a Gaussian covariance, and then used in the SLAM filter. To ensure an accurate fit, this ellipse must enclose the entire locus of the distribution. This locus is generated by moving the 2-d Gaussian distribution generated by translational uncertainty, $\Sigma_{x,y}$ through the arc generated by the uncertainty of Σ_θ. In order to accurately represent the shape with a high degree of probability, a 3σ locus is

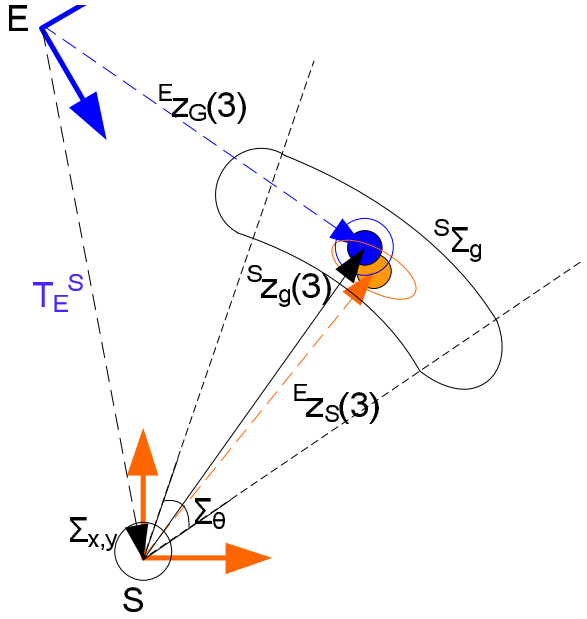


Figure 3.7: Nonlinear uncertainty from transformation. This uncertainty locus shown in black takes into account both the measurement uncertainty in the **E** frame, the translational uncertainty of the transformation $\Sigma_{x,y}$ and the transformation's rotational uncertainty, Σ_θ

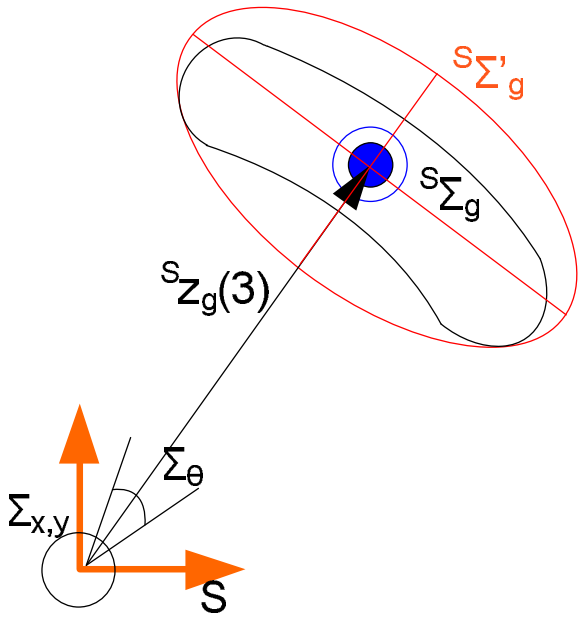


Figure 3.8: Fitting an ellipse to the non-linear PDF

used and an ellipse fit as shown as ${}^{S}\Sigma'_{g}$ in Figure 3.8. This ensures a 99.7% overlap. This is conservative, but allows the use of measurements in a linear Gaussian SLAM filter. The concept of passing a Gaussian distribution through a non-linear function and fitting a Gaussian distribution to the output is similar to the techniques used as the foundations of the Unscented Kalman Filter (UKF) [19].

3.5.1 Ellipse Fitting

This section presents the procedure to fit a ellipse around the non-Gaussian PDF generated in the last section. This is one implementation of several possible to propagate the covariance through the transformation. It works by computing a bounding box for the locus generated by moving the 3σ Cartesian uncertainty ellipse through the 3σ angular uncertainty. As discussed above, we calculate the ellipse based on the 3σ locus to ensure sufficient coverage.

Figure 3.9 shows the problem setup. Instead of operating at the (x, y) position of the

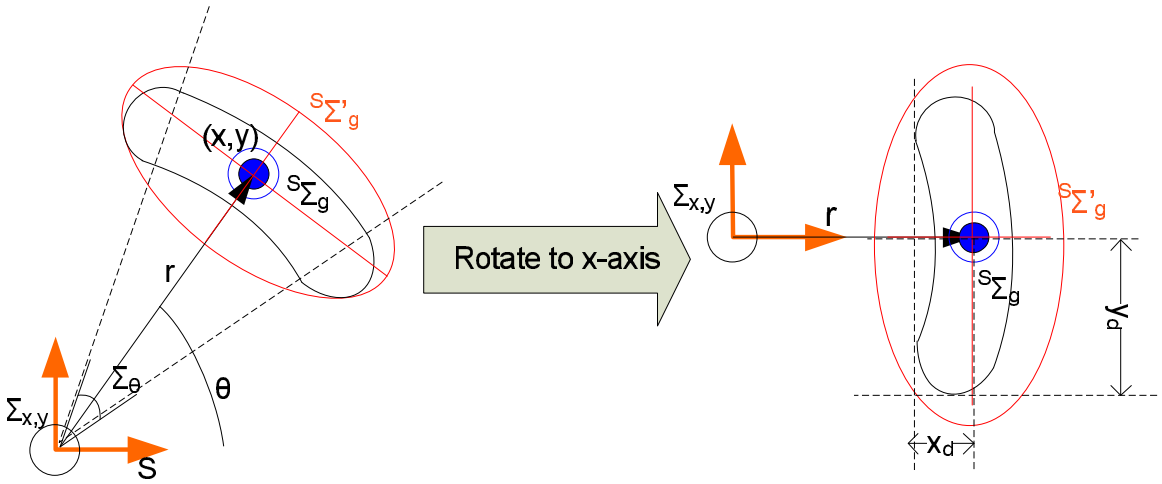


Figure 3.9: Ellipse fitting technique. A rectangle is fit around the locus traced out by the 3 covariance parameters. To simplify the procedure, the locus is first rotated to the origin, the ellipse calculated and then rotated back.

transformed mean, we rotate the locus to be centered on the x-axis, while preserving the radial distance from the origin. This simplifies the computation of the ellipse shape, by making the bounding box edges parallel with the x and y axes. Once the bounding ellipse is calculated it is rotated back to the location of the mean.

To compute the bounding rectangle we use the diagonal elements of the transformation matrix Σ_T:

$$\Sigma_T = \begin{bmatrix} \sigma_x^2 & \sigma_{xy}^2 & - \\ \sigma_{xy}^2 & \sigma_y^2 & - \\ - & - & \sigma_\theta^2 \end{bmatrix} \tag{3.19}$$

The "$-$" elements denote "don't care" elements. Ignoring the "don't care" elements of the covariance matrix as shown simplifies the problem and provides an over-estimate of uncertainty. The maximum distance from the origin is computed in the x and y directions as shown in Figure 3.9. The corners of the bounding box are found according to:

$$x_d = r - r * \cos(3\sigma_\theta) + 3(\sigma_x + |\sigma_{xy}|) \tag{3.20}$$

$$y_d = r * \sin(3\sigma_\theta) + 3(|\sigma_{xy}| + \sigma_y) \tag{3.21}$$

where:

$$r = \sqrt{x^2 + y^2} \tag{3.22}$$

where x and y are centre of the distribution as shown in Figure 3.9.

Next, the major and minor axes of the ellipse are calculated, labeled a and b, respectively. The minor axis is arbitrarily chosen to be twice the x_d distance. The size of the ellipse is given by:

$$b = 2 * x_d \tag{3.23}$$

$$a = \sqrt{\frac{y_d^2}{\frac{1-x_d^2}{b^2}}} \tag{3.24}$$

The scalar value 2 used to compute b is an arbitrary value which gives an ellipse of a reasonable aspect ratio, since the solution to this problem is a family is ellipses. In covariance matrix form, this ellipse is represented by:

$$\Sigma_g = \begin{bmatrix} (\frac{a}{3})^2 & 0 \\ 0 & (\frac{b}{3})^2 \end{bmatrix} \tag{3.25}$$

We divide the major and minor dimensions a and b by 3 to convert back to scale their magnitude back to 1σ before squaring that standard deviation to convert to variance. This ellipse must now be rotated away from the x-axis and back to the location of the transformed mean ${}^S z_g$. This is accomplished with:

$${}^S\Sigma_g = R(\theta)\Sigma_g R(\theta)^T \tag{3.26}$$

$$= R(\theta) \begin{bmatrix} (\frac{a}{3})^2 & 0 \\ 0 & (\frac{b}{3})^2 \end{bmatrix} R(\theta)^T \tag{3.27}$$

$$\tag{3.28}$$

where:

$$R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{3.29}$$

This yields an uncertainty, ${}^S\Sigma_g$ which is in the SLAM coordinate frame **S** and which is has a Gaussian distribution. This can now be used directly by the SLAM filter.

3.6 Submap Constraint Satisfaction

In much the same way CRSF SLAM uses a constraint satisfaction technique to improve the transformations between adjacent submaps, here a method is devised to apply constraints between any two submaps, which performs a weighted adjustment of the transformations of all maps which connect them.

This is used in this system when using a pair of EL transformations to refine the submap transformations calculated with SLAM. If GPS is used as the EL system, these can be referred to as *GPS constraints*. Also, during loop closure, when the vehicle's trajectory takes it through a series of new submaps and back to a known location, the the inter-map feature constraints can be used to correct the submap transformations along the sequence of maps in the loop. Applying feature constraints between adjacent maps is a special case of the algorithm presented here.

EL constraints can be used to aid the system if EL is unavailable in some submaps but then regained at a later time. An update can be performed to take into account the localization information available at either end of the submap sequence. This correction step must take into consideration the relative displacement measured between submaps as well as the fitted transformation to the external coordinate system for maps for which that is available. This is similar to the techniques used in the ATLAS framework, developed by Bosse [7].

Figure 3.10 gives a sample scenario. In this figure, an EL coordinate system is shown in blue, and 4 submap frames are shown in orange. Submaps 1 and 4 have successfully been fit to the EL frame and those transformations are indicated as ${}^{E}z_1$ and ${}^{E}z_4$, with the corresponding uncertainties ${}^{E}\Sigma_1$ and ${}^{E}\Sigma_4$.

From the SLAM process, measurements between each of the submap frames are available. These are shown as ${}^{1}z_2$, ${}^{2}z_3$ and ${}^{4}z_3$, with covariances ${}^{1}\Sigma_2$, ${}^{2}\Sigma_3$ and ${}^{3}\Sigma_4$, respectively. As with the transformations presented in previous sections, each transformation consists of a translation and rotation. So, ${}^{n-1}z_n = ({}^{n-1}z_{n,x}, {}^{n-1}z_{n,y}, {}^{n-1}z_{n,\theta})$. In order to perform an update in Kalman filter fashion, we require a way to define our system state, an observation function for measurements of the system and a method to predict measurements from the

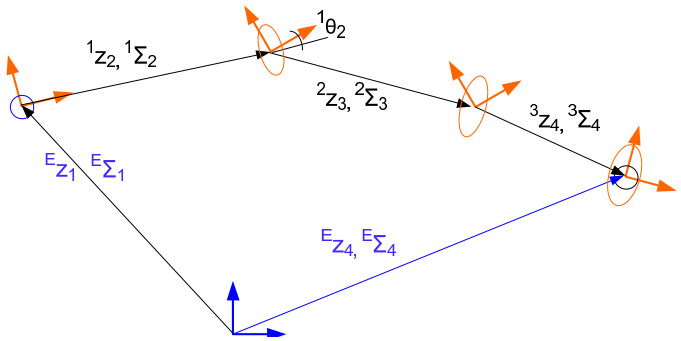


Figure 3.10: Submap constraint satisfaction

system state. Performing vector math for this example, we note:

$$^{E}z_4 - {}^{E}z_1 = {}^{1}z_2 + {}^{2}z_3 + {}^{3}z_4 \tag{3.30}$$

If we define a system state which consists of the measurements between submap frames. Given n submaps we get:

$$\mathbf{x} = \begin{bmatrix} {}^{1}z_2 \\ {}^{2}z_3 \\ \vdots \\ {}^{n-1}z_n \end{bmatrix} = \begin{bmatrix} {}^{1}z_{2,x} \\ {}^{1}z_{2,y} \\ {}^{1}z_{2,\theta} \\ {}^{2}z_{3,x} \\ {}^{2}z_{3,y} \\ {}^{2}z_{3,\theta} \\ \vdots \\ {}^{n-1}z_{n,x} \\ {}^{n-1}z_{n,y} \\ {}^{n-1}z_{n,\theta} \end{bmatrix} \tag{3.31}$$

This extends to any number of frames. The Kalman filter measurement update computes and innovation, y according to:

$$y = z - h(\mathbf{x}) \tag{3.32}$$

where z is a measurement of the system state and $h(.)$ is the observation function which converts the system state $\mathbf{x}$ into a predicted measurement. Referring back to Equation 3.30,

we assign the left-hand side as the measurement, and the right-hand side as the prediction, yielding:

$$z = {}^{E}z_n + {}^{E}z_1 \tag{3.33}$$

$$h(\mathbf{x}) = {}^{1}z_2 + {}^{2}z_3 + {}^{3}z_4 + \ldots + {}^{n-1}z_n \tag{3.34}$$

$$\tag{3.35}$$

Arbitrarily we choose to operate in the external **E** frame. It would be equivalent to operate in either frame. This gives a measurement equation z of:

$$z = \begin{bmatrix} R({}^{E}z_{1,\theta})^T \left(\begin{bmatrix} {}^{E}z_{n,x} \\ {}^{E}z_{n,y} \end{bmatrix} - \begin{bmatrix} {}^{E}z_{1,x} \\ {}^{E}z_{1,y} \end{bmatrix} \right) \\ {}^{E}z_{n,\theta} - {}^{E}z_{1,\theta} \end{bmatrix} \tag{3.36}$$

The uncertainty for this measurement is defined as:

$${}^{1}\Sigma_z = R({}^{E}z_{1,\theta})^T({}^{E}\Sigma_1 + {}^{E}\Sigma_n)R({}^{E}z_{1,\theta}) \tag{3.37}$$

Next, we must define the measurement function $h(.)$ from Equation 3.34. Since Equation 3.33 gives the transformation from Frame 1 to Frame n, $h(.)$ must provide an estimate of the same. In the given example, we wish to obtain ${}^{1}z_4$, or in general ${}^{1}z_n$. Working in the i^{th} frame, to convert a measurement from the $i+1$ frame to the $i+2$ frame, to the current frame i we compute:

$$r(i, i+2) = {}^{i}z_{i+2} = \begin{bmatrix} {}^{i}z_{i+1,xy} + R({}^{i}z_{i+1,\theta})^{i+1}z_{i+2,xy} \\ {}^{i}z_{i+1,\theta} + {}^{i+1}z_{i+2,\theta} \end{bmatrix} \tag{3.38}$$

We apply Equation 3.38 recursively to obtain ${}^{1}z_n$. This yields a measurement function $h(.)$ of:

$$h(\mathbf{x}) = {}^{1}z_n = r(1, r(2, \ldots, r(n-2, n))) \tag{3.39}$$

To perform the Kalman update filter step, we require the Jacobian $\bigtriangledown H$ of the measurement function, $h(.)$. This is more interesting to compute since the measurement function is

recursive. The Jacobian becomes:

$$\bigtriangledown H = \frac{\partial h}{\partial x} = \left[\begin{array}{c|c|c} \mathbf{I}_{2x2} \quad \partial R({}^1z_{2,\theta})^2 z_n & \cdots & \\ \mathbf{0}_{1x2} \quad 1 & & \\ \hline \cdots & \prod_{1\le k<i} R({}^{k-1}z_{k,\theta}) \quad \mathbf{0}_{1x2} & \prod_{1\le k<i} R({}^{k-1}z_{k,\theta})\partial R({}^1z_{2,\theta})^i z_n \quad 1 & \cdots \\ \hline & \cdots & \prod_{1\le k<n} R({}^{k-1}z_{k,\theta}) \quad \mathbf{0}_{1x2} & \mathbf{0}_{2x1} \quad 1 \end{array}\right] \tag{3.40}$$

where $\mathbf{0}_{m\times n}$ is a zero array of m rows and n columns, $\mathbf{I}_{m\times n}$ is an identity matrix of the same dimensions and:

$$\partial R(\theta) = \frac{\partial R(\theta)}{\partial \theta} = \begin{bmatrix} -\sin(\theta) & \cos(\theta) \\ -\cos(\theta) & -\sin(\theta) \end{bmatrix} \tag{3.41}$$

With the above equations, one can perform the Kalman update step as usual. The covariance of the state $\mathbf{x}$ is Σ_x is simply the combination of all the submap covariances:

$$\Sigma_x = \begin{bmatrix} {}^1\Sigma_2 & \mathbf{0}_{3x3} & \cdots & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & {}^2\Sigma_3 & \cdots & \mathbf{0}_{3x3} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{3x3} & \mathbf{0}_{3x3} & \cdots & {}^{n-1}\Sigma_n \end{bmatrix} \tag{3.42}$$

3.7 Extensions

3.7.1 Principle of Measurement Reproduction

Before a fitting is complete, there may be the desire to use some position information from the EL system. For example, though a single GPS measurement is not useful in a local coordinate system SLAM map, two measurements can be compared to compute a linear distance, which can be incorporated into the time propagation step of the SLAM filter.

These measurements, however, will also be used during the map fitting process, and later for localization in the SLAM process. If data is reused in a Kalman filter process

this, in effect, gives a disproportionately higher confidence to those measurements, thereby overestimating certainty. Several techniques have been developed to combine measurements in an estimator in the presence of unknown correlation. One such technique is known as the covariance intersect [18]. This can be used to prevent an overconfident estimate. However, there is significant computational complexity associated with this technique as the covariance matrix grows in size.

If measurements are reused, and the decision to do this is done in advance of their use, the Principle of Measurement Reproduction can be applied adding very little overhead [34]. This principle is a base concept of Federated Filtering [8]. Measurements used in a Kalman filter are assumed to be independent and uncorrelated. Duplicating a measurement violates this requirement. However, the filter uses the uncertainty of each measurement for weighting. This means that the total system information can be divided among several filters or several measurements [8]. A single measurement with mean μ and an uncertainty of Σ_F can be divided into n parts, each with a mean of $\mu_n = \mu$ and a covariance as follows:

$$\Sigma_F^{-1} = \Sigma_1^{-1} + \cdots + \Sigma_n^{-1} \tag{3.43}$$

where:

$$\Sigma_n^{-1} = \beta_n \Sigma_F^{-1} \tag{3.44}$$

Equivalently,

$$\Sigma_n = \beta_n^{-1} \Sigma_F \tag{3.45}$$

To ensure the "conservation of information", the fractions of information given to each measurement must sum to 1, such that:

$$\sum_{j=1}^{n} \beta_j = 1 \tag{3.46}$$

Though this is not directly a part of the fitting process flow being presented, it should be understood that this process must be applied anytime that measurement reuse is required.

Chapter 4

Experiments and Results

This chapter provides the details of the experimental setup used to test the algorithms presented in the previous chapter. The experimental procedure is outlined and the particulars of the system used to collect the data are discussed. Details of the implementation are presented and the performance of the system is evaluated.

4.1 Experimental Procedure

An approach to compare the performance of different Simultaneous Localization and Mapping (SLAM) algorithms is to test them with a data set used by other researchers in the SLAM community and characterizing the following:

- Trajectory Accuracy
- Map Accuracy
- Computational Complexity.

In many instances, SLAM is used to map the environment because other methods of localization are unavailable [35]. This means that in much of the SLAM experimental literature, truth data is not available for either the map features or vehicle trajectory. The Extended Kalman Filter (EKF) SLAM filter is often used as a benchmark for comparison.

The experiment summarized here shows how submaps can be used to:

- improve trajectory accuracy when referenced to an External Localization (EL) frame;
- maintain accurate landmark positions in the submaps; and
- reduce computation complexity, by limiting map size.

4.2 Experimental Setup

In order to evaluate the performance of the algorithm presented, the system is tested on the Victoria Park data set[1]. This data set has been used in the evaluation of many different SLAM algorithms and is considered a standard [27, 31, 20].

The Victoria Park data set was created by driving a pick-up truck equipped with various sensors through Victoria Park in Sydney, Australia. This data is provided courtesy of Jose Guivant and Edward Nebot at the Australian Centre for Field Research (ACFR) at the University of Sydney. An aerial photograph of the area is shown in Figure 4.1. Overlayed on top, in yellow, is the trajectory of the vehicle as measured by the noisy Global Positioning System (GPS) receiver. The vehicle travels the same road repeatedly in various portions of the test, so several loops can be seen.

The vehicle is shown in Figure 4.2. It is equipped with a GPS receiver, laser scanner and an encoder on the rear axle to measure vehicle speed.

The Victoria park dataset consists of approximately 60,000 odometry data points, gathered at 25 msec intervals. The duration of the entire test is just under 26 minutes. The laser scanner collected over 7,000 scans, each consisting of 360 range-bearing measurements at approximately 200 msec intervals. The GPS collected data at 250 msec intervals, but there were numerous occasions of drop-out, some of nearly 1 minute in duration. A total of about 4400 GPS position samples were gathered.

4.2.1 Vehicle Model

The encoder located on the back axle measures the speed of rotation. Based on the size of the wheels and tires used, the velocity of the vehicle can be determined. In order to

[1] available at: http://www-personal.acfr.usyd.edu.au/nebot/victoria_park.htm

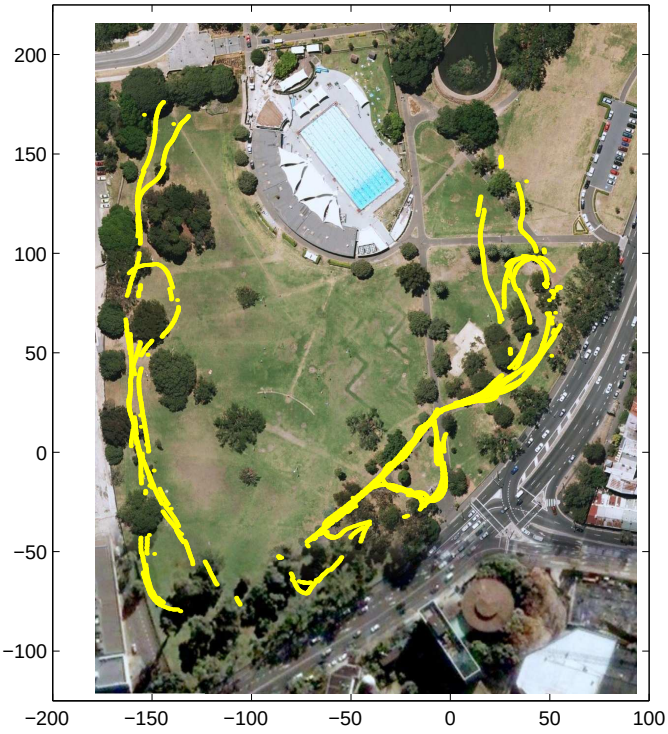


Figure 4.1: Aerial photo of Victoria Park, Sydney, Australia. This photograph was obtained by manually stitching together a series of high resolution photos from Google Maps [2]. Shown in yellow is the vehicle's trajectory as measured by the GPS receiver.

Figure 4.2: Data gathering vehicle. Note the location of the GPS antenna and laser scanner on the front bumper of the truck and the rotary encoder on the rear left wheel.

determine an accurate representation of vehicle motion from this rotational speed, a vehicle model is required.

The truck used for collecting data uses an Ackerman steering system. This is common on most passenger cars. The rear wheels are fixed and the front wheels can pivot. When the front wheels are turned, the vehicle pivots about a point along the axis of the rear axle. The development of this model can be found in [32].

The vehicle's position is defined by three variables (x, y, θ). Based on the Ackerman model, the position of the centre of the vehicle's rear axle can be calculated. Figure 4.3 shows the vehicle and parameters of the vehicle model.

For the vehicle used for this test, the parameters have the values given in Table 4.1. The motion of the vehicle is governed by a set of differential equations, which specify the location and orientation of the origin of the vehicle:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v_c \cos(\theta) \\ v_c \sin(\theta) \\ v_c \tan(\alpha) \end{bmatrix} \tag{4.1}$$

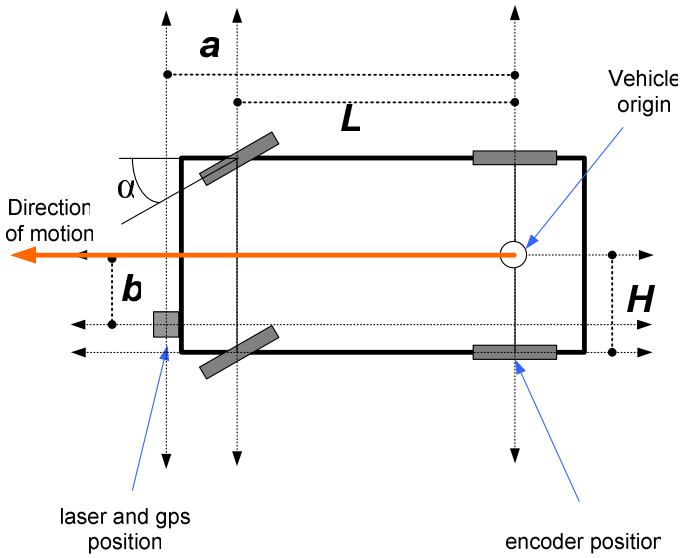


Figure 4.3: Schematic of the data gathering vehicle. The origin of the vehicle is located in the centre of the rear axle.

Table 4.1: Vehicle model parameter vales

Parameter	Value
a	3.78 m
b	0.5 m
H	0.76 m
L	2.83 m

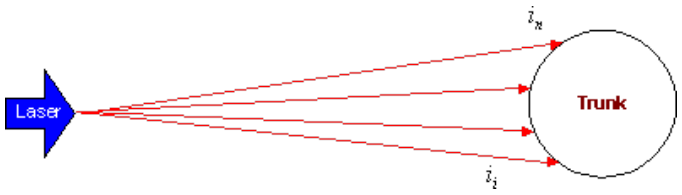


Figure 4.4: Laser rangefinder return pattern for a cylindrical object. This is a top view, showing the laser source and a cross section of a tree's trunk. Source [16].

where α is the steering input angle to the vehicle as shown in Figure 4.3, v_c is given by:

$$v_c = \frac{v_e}{(1 - tan(\alpha)\frac{H}{L})} \tag{4.2}$$

and where v_e is the speed as measured by the encoder on the rear left wheel to calculate v_c at the origin.

Measurement noise for the vehicle motion measurements is assumed be constant. A constant value of $\sigma_{v_e} = 0.1m/s$ is used for encoder velocity measurements and $\sigma_\alpha = 3°$ is used for steering angle inputs. A more complex model could incorporate high order effects, such as wheelslippage [32].

4.2.2 Tree Detection

The Gaussian SLAM process requires the use of discrete features to create a map of the environment and uses that map for localization. In this experiment, that map consists of "tree features" collected by the laser scanner.

Figure 4.4 shows an example of the expected return from the laser scanner by a tree. The laser provides range and bearing returns. By modeling the tree as a simple cylinder, the filter estimates the radius of the circumference that approximates the trunk of the tree and the tree centre position. The details of this algorithm are presented in [15] and will not be covered here.

An implementation of the tree detection algorithm written in Matlab was provided by Jose Guivant on the ACFR website and was used directly in this experiment [1] .

Measurement error for the laser scanner has range and bearing components. The LMS 200 laser scanner, manufactured by SICK, provides a 180° field of view, and samples ranges at 0.5° intervals. The laser scanner also has a maximum range of 8 m and any measurements beyond this distance are considered "no reflection." The manufacturer claims an accuracy of 1 mm within the 8 m range.

The tree detection implementation used does not provide uncertainty estimates for the range and bearing to the centre of the tree feature. In these simulations, these are assumed to be constant with a range deviation of $\sigma_r = 0.2m$ and bearing error of $\sigma_\theta = 5°$. These errors are larger than that of the sensor since the trees trunks being observed may not be perfect cylinders and so may not be modeled properly.

4.2.2.1 Data Association

In this implementation, there is no way to uniquely identify map features by sensor measurements alone. This means that a statistical data association technique is required. In this case, a linear search for nearest-neighbour is performed. This is computationally intensive, since each search is $O(n)$ based on n features. A more efficient implementation could be used for a real-time system.

The distance between two features is computed as the Mahalanobis distance, shown in Equations A.1. Also computed is the normalized innovation squared, which is used as a gate to determine quality of fit. More specifically, using Kalman filter notation, given a measurement z with uncertainty R, and a state x with uncertainty P:

$$y = z - h(x) \tag{4.3}$$

$$S = HPH^T + R \tag{4.4}$$

$$\tilde{I}^2 = y^T S^{-1} y \tag{4.5}$$

where $h(x)$ is the observation function, with the Jacobian H. The result, $\tilde{I}^2$ is the normalized innovation squared. This result can be compared to a constant gate to determine if an existing feature will be augmented or a new feature created. For example, if $\tilde{I}^2 < 3^2 = 9$ the features are within 3 standard deviations of one another, ensuring an overlap of their 99.9% confidence ellipses. In these simulations a gate of $\tilde{I}^2 < 9$, which represents 3 sigma is used when determining the minimum distance for association to an existing feature. In

order to create a new feature, the minimum distance from an existing feature must be 5σ, or $\tilde{I}^2 > 25$. This gives good confidence in association and minimizes the creation of spurious features.

4.2.3 CRSF Implementation

As discussed by Williams in [35], there are several possible techniques for the implementation of Constrained Relative Submap Filter (CRSF) SLAM. There are numerous implementation details not discussed in [35], so a fair amount of design work was involved in the implementation used herein.

As presented in [35], each implementation must provide a method to do each of the following:

- track submaps and their relative transformations
- transition the vehicle from one submap to another
- apply constraints between submaps
- perform loop closure

The following 4 subsections discuss the implementation used in this simulation for each of the tasks listed above.

In order to operate the CRSF algorithm and the EL fitting process, a certain set of data is required to be associated with each map. Stored within each map structure are the following sets of data:

- Reference to parent map - this is the index of the node in the tree which is parent to this map. This is used when traversing the tree to determine relative transformations
- References to any child maps and their relative transformations
- Map size - in this implementation, all maps are of the same dimensions, 80 x 80 m. In other implementations, this size could vary from map to map. This is discussed further in Section 4.2.3.2

- Vehicle and feature states and covariances - the most recent estimate of the features within the map and the vehicle's current position
- Vehicle trajectory - a stored version of all vehicle position estimates. This is used for display purposes.
- EL transform estimate - an estimate of the submaps position in the external coordinate frame. This is not initialized until the transformation passes the user-defined quality metric.

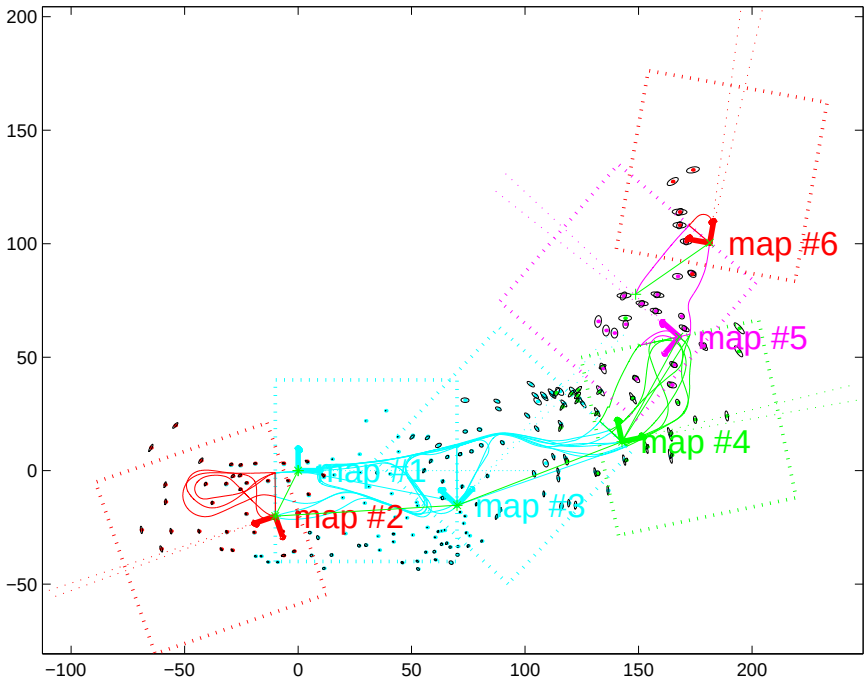
The following sections discuss the usage of this data in the CRSF SLAM process.

4.2.3.1 Map Tree

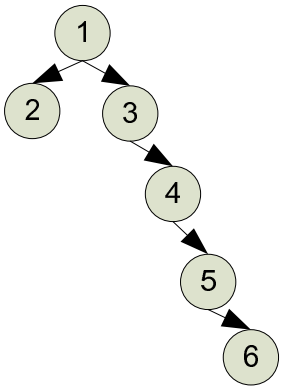
An implementation of CRSF, provided by Williams, was designed for a smaller data set, a synthetic environment and a trajectory which was manually created. With the larger data set, natural features and less accurate odometry sensors as studied in this thesis, new issues arose and required implementation changes. In this implementation, submaps are created *dynamically*. This enables the system to operate on any subset of the input data set.

A configuration with fixed map locations was not possible for this dataset. Instead, a system which dynamically created maps and determined transitions was created. A tree data structure was selected for storage of the submaps. In order to provide a links between maps, a parent-child structure was chosen, in which each node in the tree could have multiple children, but only one parent. This lead to an unbalanced tree implementation, where each map could have only one parent, but multiple children.

An example is shown in Figure 4.5. This example uses the first 12,000 samples from the Victoria Park dataset, and does not perform any global alignment. As can be seen, a total of 6 maps are created. Each map has a bounding box, shown as a dotted line. This bounding box size is predetermined, and as the vehicle leaves a map by crossing this boundary, it either re-enters an existing map or creates a new map. Changing the parameters of the algorithm, such as map size, or running a different set of data would result in a different set of submaps being created.



(a)



(b)

Figure 4.5: Map tree example. The map in (a) results in a map hierarchy tree shown in (b). This trajectory results from the first 12,000 samples of the Victoria Park dataset. Map boundaries are shown as dotted lines, with each map's origin and orientation indicated with an xy axis. The vehicle trajectories are shown as solid lines. Map features are shown dots, coloured to match their respective maps.

Determining the transformation between any two maps involves finding the path between any two nodes of the tree and combining the transformations. This is done by first determining an edge list which defines that path. Negative numbers are used to indicate a transition up the tree, and positive indicate downward. The numerical value of the transition represents the child edge being taken. For example, the path from map 2 to map 5 in the tree can be written as:

$$-1,\ 2,\ 1,\ 1$$

Once a path is found, the relative transformation between the end points is simply the combination of all transformations along the path. For negative path values, the inverse transformation is used.

When combining transformations, the uncertainties of those transformations become cumulative and are added together to determine the resultant uncertainty.

4.2.3.2 Map Re-entry

A property of each submap is its size. For these simulations, this size was fixed to a constant set of dimensions. Specifically,

$$(x_{min}, x_{max}, y_{min}, y_{max}) = (-10, 70, -40, 40) \tag{4.6}$$

This provided an 80 m by 80 m map. Based on the feature density of this dataset, this provided good feature content within each map, which provides rich enough information to prevent drift, while still capping computational complexity. In an implementation with unknown feature density, extensions could be made to determine the map dimensions dynamically.

Using the technique presented in Section 4.2.3.1 to determine relative transformations between submaps and each submap's size, the re-entry of an existing map can be detected. When exiting a submap, the system checks to see if the current position lies in any existing submaps. If not, a new map is created.

When re-entry into an existing submap is detected, the vehicle's position in that map must be reinitialized. In order to do this, the vehicle must re-associate its current sensor readings with previous portions of the map. The technique presented by Williams in [35]

uses the features in view to localize the vehicle based by comparing them to existing map features. This technique was also used in these simulatioins.

This process aids positioning since re-entering an existing map can correct the vehicle's position estimate and map features are continually refined each time a map is revisited. Since the contents of each map are independent from all other submaps, the refined feature estimates can be used to improve the relative transformation estimates between maps, as presented in the next section.

In implementation, a heuristic was required to reduce unnecessary map re-entries. A situation can arise as follows: As a vehicle exits a submap and enters an existing map, feature association is used to determine the vehicle's location in the re-entered map. Due to accumulated trajectory errors or map transformation inaccuracies from the previous submap, the result of feature association may place the vehicle's actual position outside of the re-entered map's boundaries. This would cause the vehicle to transition back to the previous submap. This process would repeat, transitioning the system rapidly between adjacent submaps. By simply setting a minimum time between map transitions, the rapid switching was eliminated.

4.2.3.3 Map Feature Constraints

A fundamental operation required in both Constrained Local Submap Filter (CLSF) and CRSF SLAM is the constrained alignment of two submaps based on their overlapping features. The details of this process were presented in Section 2.4.1. The technique, however, has been extended in this implementation.

Based on the approach presented in Section 3.6, constraints can be applied not just between maps related as parent and child, but between any submaps in the map tree. This means that any submaps containing observations of the same features can be used to refine all relative transformations between those two maps using the process presented in Section 3.6.

For example, in Figure 4.5, map 3 and map 5 are physically close maps, though map 4 resides between them in the map hierarchy tree. The map size limits only the position of the vehicle to be within the submap. Since the laser scanner can see beyond the edge of maps, features may exist in a map beyond the edges of the boundaries shown. In this

example, that means that map 3 and map 5 may contain observations of common features. It would, therefore, be advantageous to apply feature alignment constraints between these two maps.

In this implementation, each map was compared to all other maps to search for overlapping features and these feature constraints were applied to all matches.

4.2.3.4 Loop Closure

Loop closure, for this SLAM system, is simply a special case of the inter-map constraint satisfaction algorithm presented above. Loop closure is detected in the same way as the overlap of any two maps. Propagation of corrections through the map tree is performed in the same was as discussed above.

4.2.4 GPS Data

The GPS receiver located on the front of the truck measures its position in Universal Transverse Mercator (UTM) coordinates, also known as Northings and Eastings [12]. These coordinates are Cartesian and can be used with no transformation as would be required for Latitude and Longitude.

One shortcoming of the data set is the lack of a GPS quality metric. Only a position estimate is available. This requires the system to use a fixed uncertainty, and for correctness, this must be at least as great as the worst receiver position error. As such, an error with a standard deviation of $\sigma = 10m$ is used for the x and y axes for all measurements.

As can be seen in Figure 4.1, there were periods of GPS drop out during this test. Each data point was accompanied by a sample time, which is used to align the data with vehicle odometry and SLAM measurements.

4.2.4.1 Map fitting data storage

In order to determine the transform estimate of a map to the external localization system, as presented in Section 3.2, a synchronized set of SLAM trajectory data and external localization measurements must be collected. This data is collected until there is sufficient

information available to meet the uncertainty requirement specified by the implementation. Stored in each map is the matched set of trajectory data. If a fit is unable to be computed before exiting a map, this data is stored. Upon returning to the map, the data is continually added as long as external localization information is available. This means that the transformation does not have to be estimated before leaving the map, but can be estimated upon re-entry, as soon sufficient information becomes available.

4.2.5 Simulation Parameters

Both the CRSF and transform estimation algorithms have several operational parameters which are chosen at run-time. These parameters can affect the quality of the output trajectory and maps as well as the computation time required for operation.

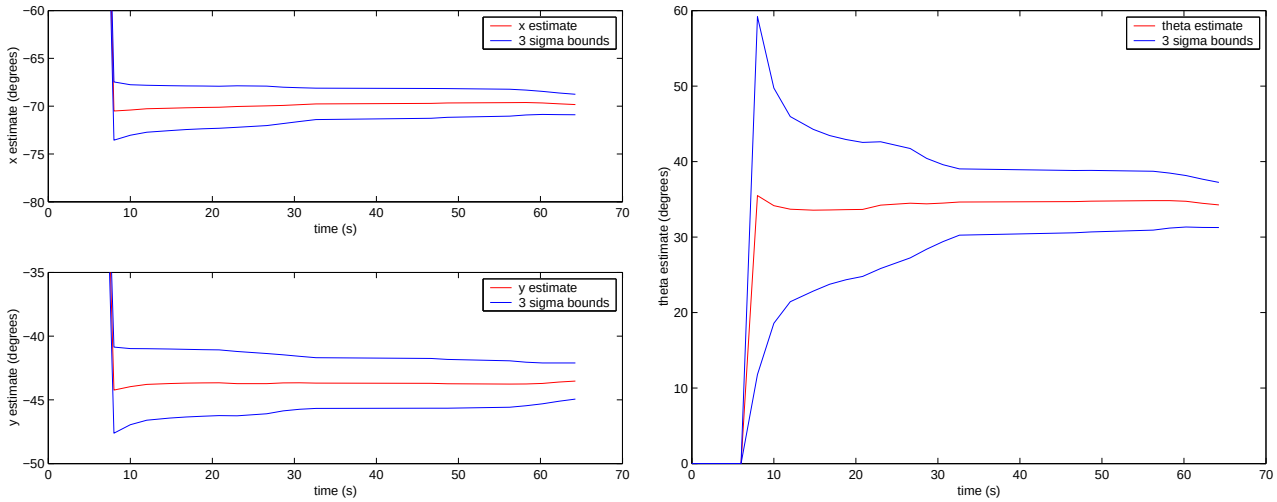
4.2.5.1 Transform Estimation Parameters

When performing map transformation estimation, the user must specify a tolerance which defines an acceptable fit. It is this gate that is applied to the transformation uncertainty estimate Σ_T developed in Section 3.3. When paired with the result of the chi-squared test, this determines when to perform the delayed initialization of the submap's transform. This gate can operate on all three variables of the transformation. Selecting a 3σ condition for the translational and rotational components allows the user to minimize the position offset and the non-linear effects of rotational uncertainty.

Figure 4.6 shows the results of a transform estimation procedure for a single map from the Victoria Park data set. This is for the first 60 seconds of data in the set. Transform estimation does not begin until approximately 8 seconds into the simulation. Since estimation with a small set of data is unlikely to yield good results, data is collected, but estimation does not begin until at least 10 GPS samples are available.

The x and y estimates converge quite quickly. The 3-sigma bounds are less than 3 m after the first estimate and after sixty seconds are reduced to approximately 1 m.

The transform uncertainty estimate is a function of GPS and SLAM trajectory uncertainties. If both sets of trajectory uncertainties were fixed at a constant value, increasing the number of samples would result in lower uncertainty. However, in a SLAM system,



(a) (b)

Figure 4.6: Transform Estimate Convergence Example. Figure (a) shows the x and y estimate convergence, while (b) shows θ.

as the vehicle explores, odometry errors compound and without returning to a known location, the vehicle's position uncertainty grows. This means that the uncertainty of the transformation estimate may reach a practical limit in implementation, hitting a point of diminishing returns, where recording additional samples does not significantly reduce the transform uncertainty.

A system designer should take this into consideration when selecting transform quality gate characteristics. Specifying a gate condition which is too aggressive could prevent the transform from ever meeting the gate condition.

For this data set, the gate condition used was:

$$3\sigma_\theta < 3^\circ \tag{4.7}$$

$$3\sigma_x < 1\ m \tag{4.8}$$

$$3\sigma_y < 1\ m \tag{4.9}$$

This yielded a condition which could be met within a reasonable amount of time, to enable GPS usage within submaps.

4.2.5.2 CRSF Parameters

The main parameter left to the system designer of a CRSF SLAM implementation is the condition which determines submap size. As presented in Section 2.4.1, this condition may specify a maximum a map area, maximum number of features, maximum allowable vehicle uncertainty or other such conditions.

If the map feature density is approximately constant, setting an area limit for a map is approximately equivalent to limiting the number of features in that map. For a real-time implementation, this is of particular concern since the number of map features determines computational complexity.

This submap area limit is enforced by defining boundaries for a map. When the vehicle crosses the edge of a map, the CRSF algorithm exits that map and moves to a new submap. As mentioned in Section 4.2.3, the implementation used here defined a maximum area of 80 m x 80 m. As will be seen later in Section 4.3.5, for the Victoria Park data set, this limits the maximum number of features in a given CRSF map to approximately 200, instead of the approximately 700 observed with full EKF SLAM filter.

4.3 Results

Both subsets and the complete Victoria Park dataset were run through simulations. The results of these experiments are presented in this section. Following William's terminology from [35], the full EKF SLAM filter, sometimes referred to as 'vanilla SLAM', will be referred to as the Absolute Map Filter (AMF). This will be used as the benchmark for comparison.

4.3.1 Graphical Map Representation

When map and trajectory results are presented together, they will be shown using an overhead map representation, like that shown in Figure 4.7. A series of 6 submaps are shown in this figure. Submaps which have a satisfactory EL frame transform are shown in blue and green, for even and odd indices, respectively. These are termed *locked* maps. Those which do not are shown in red and purple. The vehicle's trajectory, features and 1σ-feature uncertainty ellipses within each map use that map's colour. An x-y axis is shown for each map, denoting its orientation. Submap boundaries are shown with a dotted line.

The transform estimates between maps have an associated uncertainty, Σ_T, which is shown in Figure 4.7. Emanating from the origin of each submap along its x-axis are two dotted lines which represent the rotational uncertainty, Σ_θ of each submap's orientation. These represent a $\pm 1\sigma$ deviation. Located at the origin of each submap is an ellipse which indicates the xy uncertainty, Σ_{xy}.

In order to show the effect of this transform uncertainty on the feature locations, an additional 1σ uncertainty ellipse is drawn for each feature. Drawn in black is an ellipse which represents the feature uncertainty in the EL frame. Note how this uncertainty is larger for features further from the origin. Also note, that this uncertainty is small for maps 1, 3 and 4 which have satisfactory EL transformations. This uncertainty grows, larger as the maps move further from locked maps, and is largest in map 6.

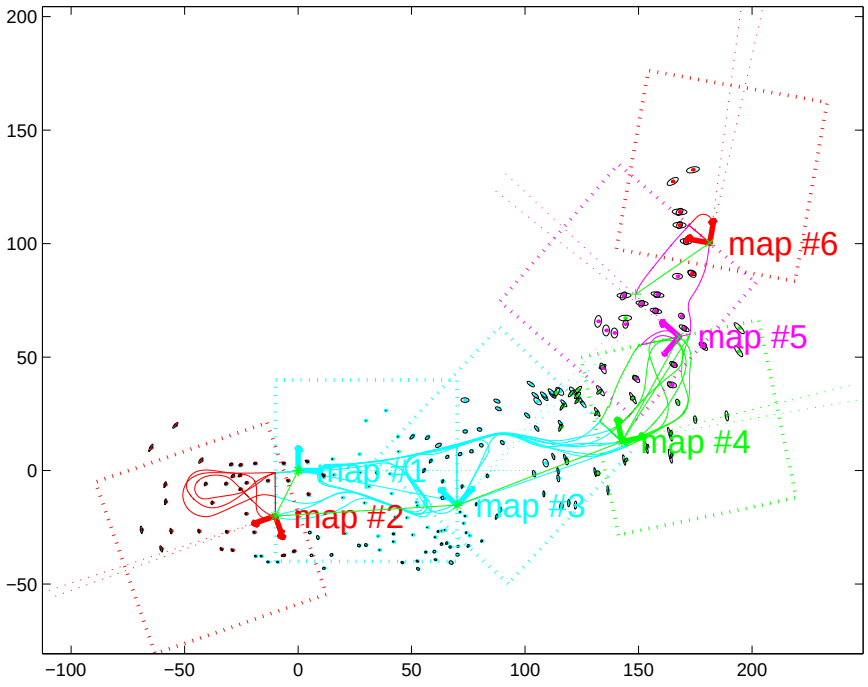


Figure 4.7: CRSF SLAM Map Example. Represented from Figure 4.5.

4.3.2 Aided AMF

The AMF is simply a special case of the CRSF SLAM filter. In this case, the map size is not limited, so only 1 map is ever created and no constraint satisfaction is performed. However, that does not mean that EL transformation estimation cannot be conducted. In fact, it can be of significant aid to the AMF.

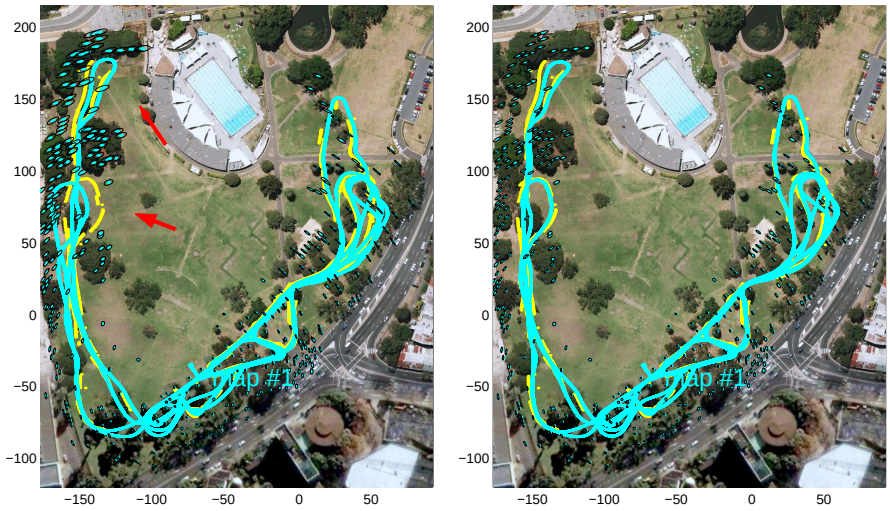
For example, in Figure 4.8, the results of the AMF on the full dataset are shown. In Figure 4.8a, the EL transformation is measured, and used for display purposes, but is not used to aid vehicle state estimation. In Figure 4.8b, once a satisfactory transformation is determined, further localization measurements are incorporated into the AMF. The red arrows indicate portions of the trajectory which differ significantly from the GPS measured trajectory, shown in yellow. At its furthest point, the AMF trajectory deviates by over 15 m from the GPS measured trajectory.

This comparison of GPS and AMF trajectory provides shows that the use of EL information can be used to correct AMF trajectory drift. Since no ground truth data is available for the map features, a quantitative comparison of feature position cannot be done. However, using the satellite photo for comparison, it can be see that improving the trajectory by using external aiding also results in more correct feature location estimates, which was a goal of the research. Also note the lower covariances for map feature estimate on the left side of the map.

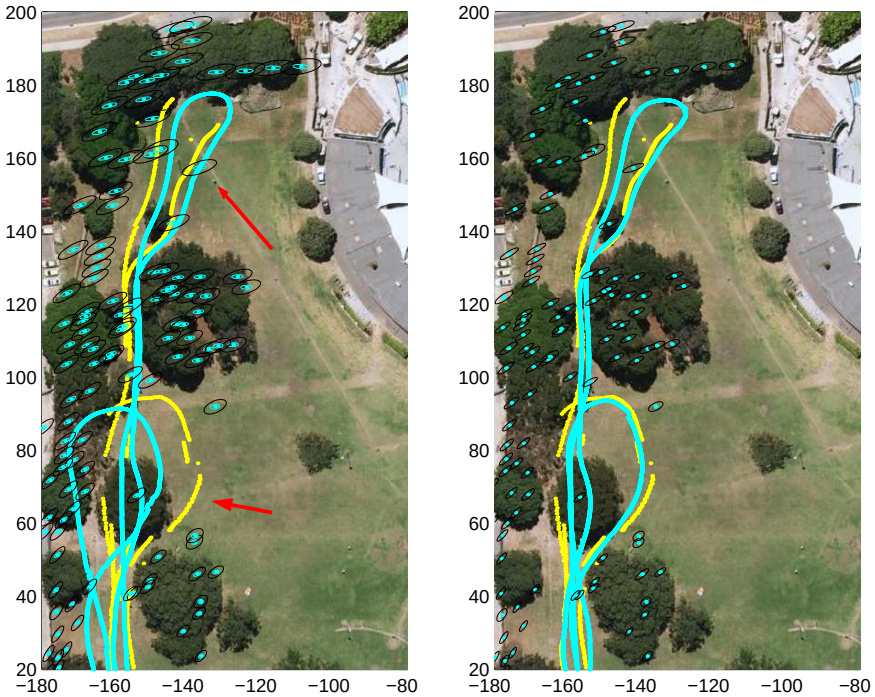
4.3.3 Trajectory Comparison

In the same way as AMF and GPS-aided AMF were compared in the previous section, here we compare CRSF and GPS-aided CRSF. With the AMF trajectory used as a reference, the results of the unaided CRSF algorithm and the external-localization-aided CRSF algorithm can be compared. The metric chosen for this comparison is the distance between the pair of trajectories, evaluated for each time step.

Figure 4.9 compares the trajectories for the first 24,000 steps. The AMF is used as the baseline for comparison. Trajectories for CRSF and GPS-aided CRSF are shown. The CRSF consists of several submaps, so abrupt changes in position can occur as the vehicle transitions between submaps. For most maps, the two CRSF trajectories are identical.



(a) Without GPS aiding (b) With GPS aiding



(c) Closeup without GPS aiding (d) Closeup with GPS aiding

Figure 4.8: EL aiding in AMF. Plot (a) shows the trajectory and map estimates for the AMF filter run on the full dataset. The areas marked with red arrows show significant drift from the GPS measured trajectory (shown in yellow). Plot (b) shows the same trajectory with GPS aiding. Plot (c) shows a closeup of (a) and plot (d) shows a closeup of (b).

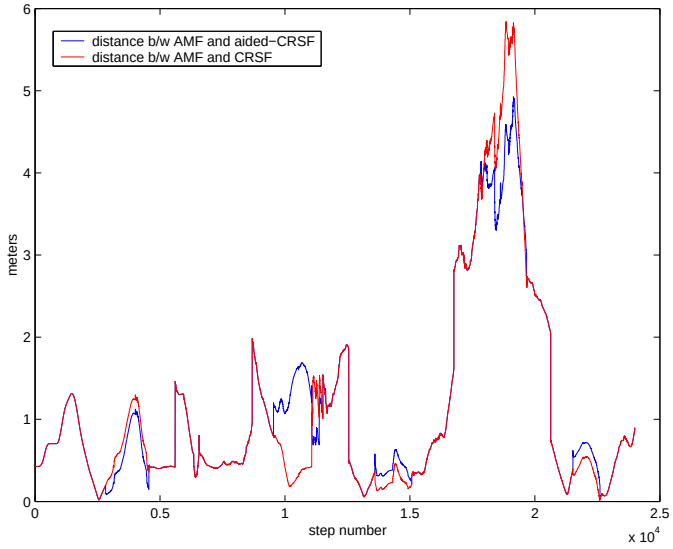


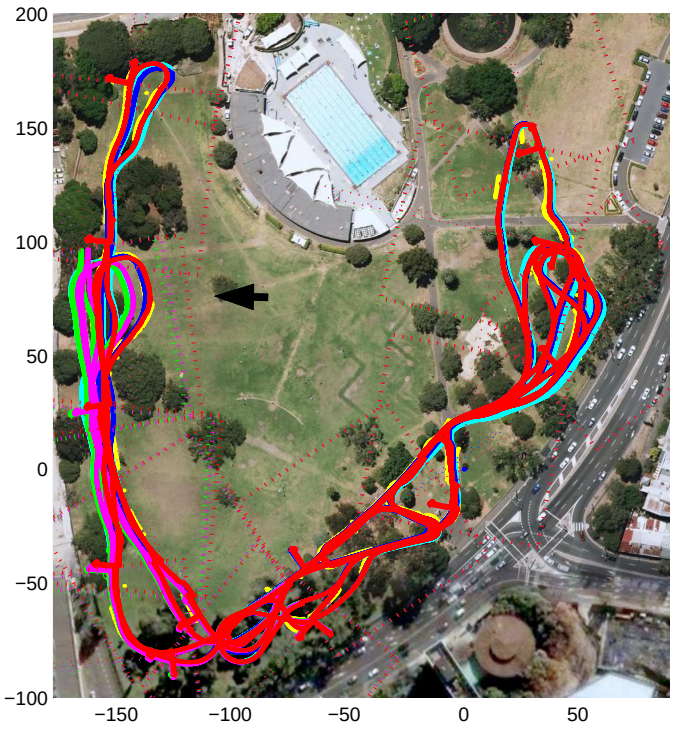
Figure 4.9: Trajectory Error Comparison. Shown are the distance distance between the CRSF and aided-CRSF trajectories the baseline AMF SLAM trajectory

This can be seen in the regions where the red and blue curves overlap.

Figure 4.10a shows the full vehicle trajectories overlayed on top of a satellite photo. Figure 4.10b shows a close-up of the portion indicated by the black arrow. With no numerical ground truth data, the AMF is compared to the GPS measured trajectory. The unaided AMF trajectory deviates from the GPS measurements. Using GPS-aiding, the trajectory more closely matches the GPS points.

The frame shown with a red axis at the top of Figure 4.10b is a locked CRSF submap, with a known EL transform. The colours for this map deviate from the standard presented in Section 4.3.1 so several filters could be compared. This map is, therefore, fixed and does not move with the application of submap constraints. The unconstrained CRSF data shows a discontinuity with this submap due to transformation drift between submaps. The application of inter-map constraints shows a smaller discontinuity. However, due to the drift of the unaided SLAM process, the result does not match measurement from the GPS. It should be noted that the unaided CRSF has a similar trajectory to the unaided AMF.

The application of EL constraints to the CRSF trajectory reduces the inter-map dis-



(a) Full Map

Figure 4.10: Full trajectory comparison. *yellow* = GPS, *cyan* = AMF, *blue* = AMF w/ GPS aiding, *green* = unconstrained CRSF, *magenta* = CRSF w/ inter-map constraints, *red* = CRSF w/ inter-map and GPS constraints

(b) Close-up

Figure 4.10: Full trajectory comparison. *yellow* = GPS, *cyan* = AMF, *blue* = AMF w/ GPS aiding, *green* = unconstrained CRSF, *magenta* = CRSF w/ inter-map constraints, *red* = CRSF w/ inter-map and GPS constraints

continuity even further. The aided-CRSF trajectory more closely matches the aided-AMF result. As will be shown in Section 4.3.4, trajectory corrections resulting from constraints also implicitly improve the position of the maps in the external frame, and therefore improve the location of the features within those maps.

4.3.3.1 Trajectory Covariance

Since the CRSF resets the state covariance to zero each time it generates a new map, the covariance cannot grow unbounded, as it can with the AMF. This means that local map features will have a lower covariance in the local submap frame, though their uncertainty, once projected into other frames, must take into account transformation uncertainty [35]. In cases where local submap accuracy is of importance, the CRSF is an ideal choice.

Figure 4.11 shows the local map trajectory uncertainties for the first 24,000 samples of the dataset. The 1σ CRSF submap uncertainties have an upper bound, since the vehicle transitions between maps on fixed boundaries. Note that the CRSF covariance is equal to the AMF covariance at the beginning of the simulation until the vehicle transitions out of the first map.

Figure 4.12 shows the covariances of the trajectory once they are converted to the EL frame. Since the transformation to this frame has some uncertainty, the resultant covariance grows. Shown are the covariances for the trajectory before and after the map constraint alignment. Note how the alignment reduces the estimate uncertainties. This is shown in sections of the figure where the blue line is lower than the green.

4.3.4 Map Feature Results

Landmark features will be compared for each of the SLAM filters. Figure 4.13 shows several close ups of map features. These are from different sections of the trajectory to provide representative samples. Figure 4.13a indicates the location of each close up.

In all cases, the AMF features have the largest local covariance. Since CRSF vehicle covariances are reset to zero at the beginning of each submap, local feature covariances remain low. The aided-AMF has lower covariances than the unaided-AMF since EL measurements from the GPS keep the vehicle covariance bounded. This is the case in all 3

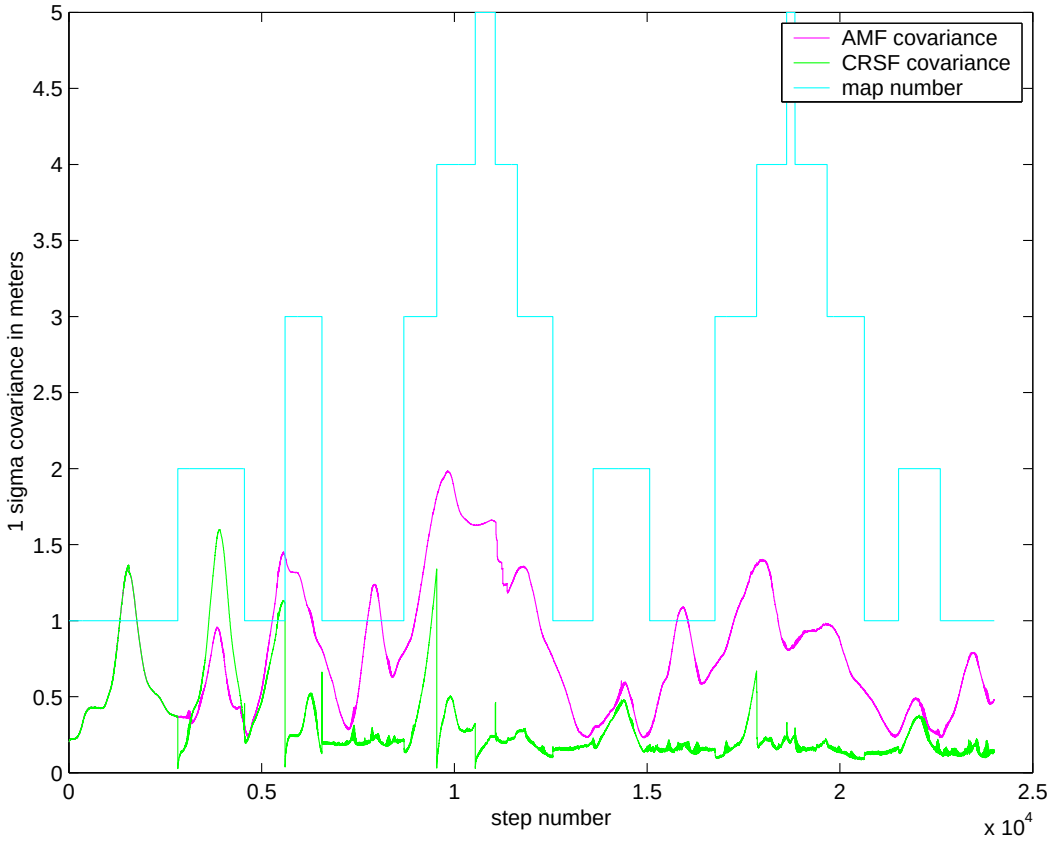


Figure 4.11: Local trajectory covariance estimates for AMF and CRSF

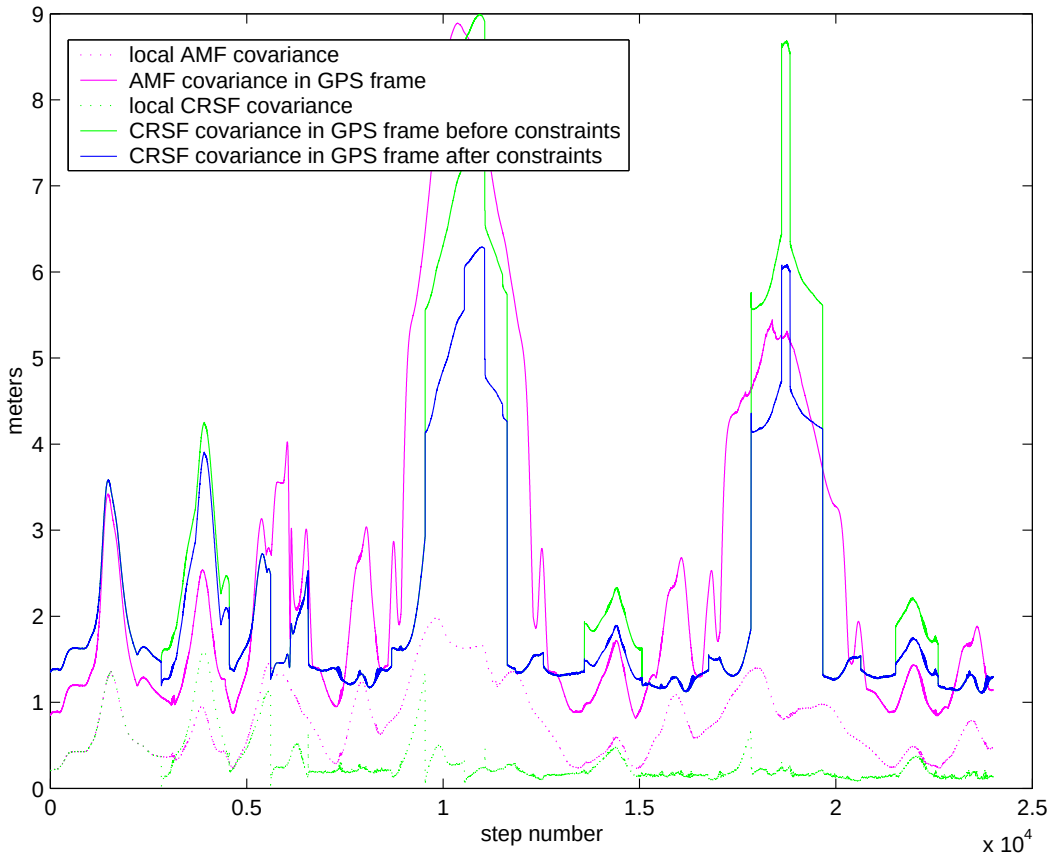
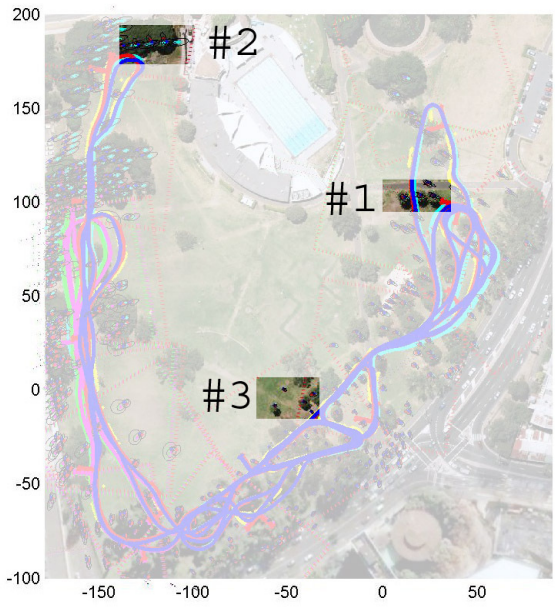
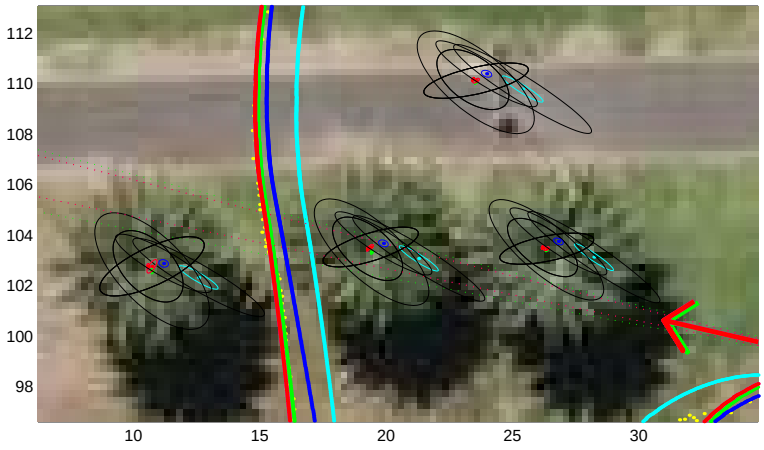


Figure 4.12: External frame trajectory covariance estimates for AMF and CRSF

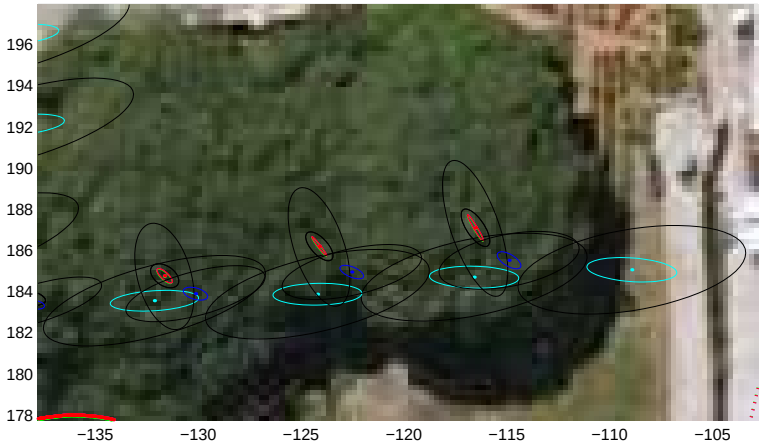


(a) Highlighted close up locations

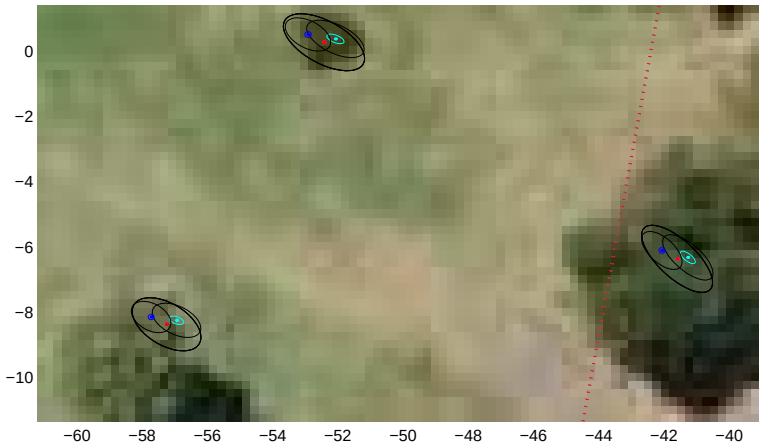
Figure 4.13: Map feature close-ups.



(b) Close up #1



(c) Close up #2



(d) Close up #3

Figure 4.13: Map feature close-ups (continued). *yellow* = GPS, *cyan* = AMF, *blue* = AMF w/ GPS aiding, *green* = unconstrained CRSF, *magenta* = CRSF w/ inter-map constraints, *red* = CRSF w/ inter-map and GPS constraints

Table 4.2: Filter computation times

Filter	Run-time (seconds)
AMF	3135.5
AMF w/ GPS aiding	3645.4
CRSF SLAM w/ GPS Aiding	1496.7
Submap alignment post-processing	93.17

close up examples.

The covariances in the EL frame do not follow the same trend. The AMF has larger map covariances since its vehicle covariances grow, and those propagate to the EL frame directly through the estimated EL transform. The unconstrained CRSF covariances are large, but after feature constraints are applied, the transformations between maps are refined and this uncertainty shrinks. Though there is no numerical ground truth data, it can be seen that the filters which use GPS aiding have feature locations which are closer to the centres of the features in the satellite photo. This system, therefore, accomplishes the goal of using EL measurements to eliminate the drift seen in the AMF algorithm.

In some cases, features of some colours are not visible. This occurs if there is exact overlap with a feature from another filter.

4.3.5 Computation Time Comparison

The simulations were carried out entirely in Matlab, as previously mentioned. A more efficient implementation could be written using a compiled language such as C. No efforts were given to obtain real-time results, but it has been shown in other work, such as by Williams in [35], that real-time implementations of the CRSF SLAM algorithm is possible if the filter parameters are chosen appropriately.

Computations were done on a AMD Athlon XP 1600+ processor with 1 gigabyte of RAM. This system was used for all tests in order to provide comparative timing measurements. Table 4.2 gives the computation times required to run each of the SLAM algorithms being tested.

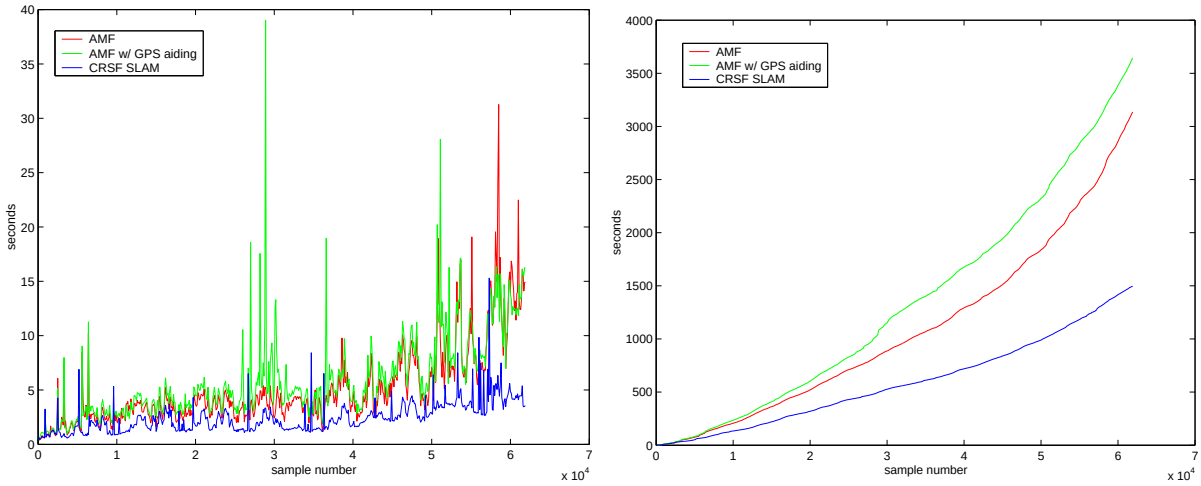
It should be noted that in this implementation the inter-map feature alignment constraints and GPS constraint processes are running as a post-processing task. In a system which requires this process to be carried out in real time, the inter-map alignment can be computed for maps which are presently inactive. Calculations can be carried out whenever free cycles become available and the update applied when the process is finished, since this step is independent of the map building process. Table 4.2 show that this map alignment procedure adds less than 7% overhead to the CRSF SLAM process.

The implementation of this inter-map alignment is done by performing an alignment between each map and all others. An increase in efficiency could be gained by limiting this to only maps in the vicinity.

To further compare run times, Figure 4.14 compares the CPU time per 100 steps. The AMF with GPS aiding requires the most computations. In order to obtain an accurate transformation, map fitting must be performed numerous times until a satisfactory result is obtained. The tight tolerances of $3\sigma < 3^\circ$ used in this experiment mean that a fit is not obtained until around sample 30,000. This can be seen in Figure 4.14a. After the submap is locked to the external reference frame, the computation rate matches the AMF more closely. This can be seen in Figure 4.14b since the red and green lines grow at the same rate after this point.

In both plots, the CRSF implementation runs faster, using less time per 100 steps. This results from the limited submap size. This is a significant advantage of the CRSF. As seen in Figure 4.14a, the complexity of each step grows very little with time, whereas the AMF increases exponentially as the map grows. This achieves the goal of maintaining the bounded complexity of the CRSF SLAM approach, while aiding in the use of EL.

Figure 4.15 shows the number of active features being processed at each time step. The AMF filters grow continuously as new features are added to the map. The CRSF maps do not grow beyond a fixed value, since the map sizes are limited. Because the maps are limited by physical size, as opposed to the number of map features, the number of features in each map can differ.



(a) Run time per 100 steps (b) Cumulative run times

Figure 4.14: Run time comparison for AMF, AMF w/ GPS aiding and CRSF

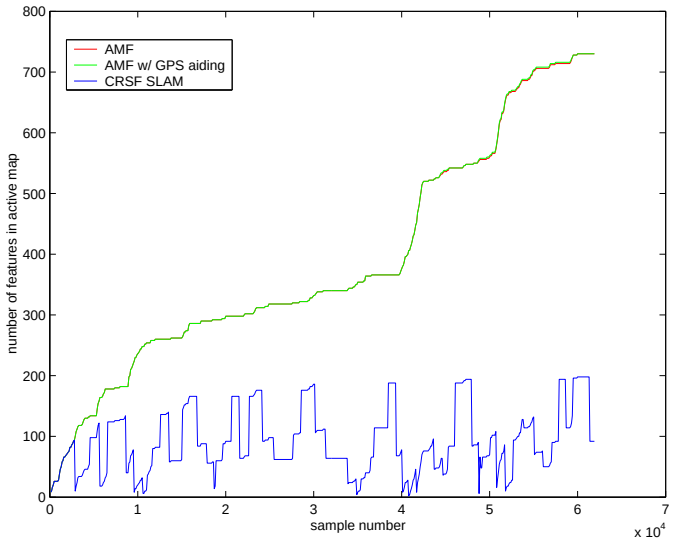


Figure 4.15: Active feature comparison for AMF, AMF w/ GPS aiding and CRSF.

4.3.6 Memory Usage

Memory usage for the MATLAB implementations tested was not evaluated. Storage requirements for a SLAM system are based on state and covariance storage requirements. State storage grows $O(n)$ with the number of map features n, while the covariance grows with $O(n^2)$. In order to represent the full set of features in a single map, all the algorithms evaluated require space for this map and its uncertainties.

The CRSF SLAM algorithm, requires additional space to store local submaps. These submaps have sizes proportional to the map above. However, they have fewer features. For a submap with m feature, the state and covariance sizes are $O(m)$ and $O(m^2)$, respectively. A system consisting of p submaps will require $O(pm)$ and $O(pm^2)$ for state and covariance storage. It should be noted that compared to the traditional SLAM implementation above, $pm \geq n$. This inequality arises since some overlap exists between maps, so features will be represented more than once.

Computing a transformation from a local submap to an external reference frame requires the additional estimation of only the 3 state variables, and the corresponding $3x3$ covariance estimate, since this estimate is independent of the vehicle trajectory within the local map and that maps features.

Chapter 5

Conclusions

Summarized in this chapter are the goals of the research and the performance of the algorithms presented. The key contributions are highlighted and some suggestions for future work are made.

5.1 Goals

As presented in the first chapter, the goals of this research are:

1. to provide a method to supplement a computationally efficient Simultaneous Localization and Mapping (SLAM) implementation with the ability to utilize External Localization (EL) information, when available; and

2. use that information to correct previous trajectory and feature location estimates made when external localization measurements were not available.

This thesis has shown that it is possible to use EL systems within both the Absolute Map Filter (AMF) and Constrained Relative Submap Filter (CRSF) SLAM environments. Increased positional accuracy was shown for both vehicle trajectories and map features by removing the accumulated position drift and uncertainty associated with unaided SLAM systems.

The data set used included EL data which was noisy and contained periods of dropout. The algorithms presented proved tolerant to those conditions. In addition, at times

when EL measurements were regained after a drop-out, corrections were made to previous submap transformations, thereby correcting vehicle trajectory and landmark map estimates for which EL was unavailable.

It was shown that the CRSF SLAM implementation involves less computational complexity than the AMF through the reduction of the number of active features processed at any one time. EL aiding added less than 7% overhead to the CRSF process, which still required less than half the computation time of the AMF SLAM system. This, therefore, accomplished the goal of supplementing a computationally efficient SLAM filter.

5.2 Contributions

Several contributions to the field were made as a result of this research. This thesis presented a novel approach to integrating EL with SLAM. Through the minimization process given in Chapter 3, a transformation between the SLAM and EL frames can be estimated. Non-linearities can by avoided by using a quality metric to delay the use of this transformation.

This basic structure should be compatible with relative localization systems other than Extended Kalman Filter (EKF) and CRSF SLAM. For example, a system using radio beacons may also use other EL measurements, such as Global Positioning System (GPS), with no *a priori* knowledge of the relationship between the coordinate frames. Additionally, other localization systems and SLAM algorithms could benefit from this use of the techniques presented in this thesis.

For implementation simplicity, some approaches were taken which created some weaknesses with the implementation. For example, a fixed map density was assumed and, therefore, fixed map dimensions were selected. In a real-time implementation, this could result in a system that could exceed its specified time constraints. To improvde this, alternate approaches could dynamically size maps based on their complexity.

More accurate results could be obtained with better models of the vehicle and the sensors. A specific shortcoming of the simulations in this thesis was the use of static GPS uncertainties, due to the lack of GPS quality data. This data is available from most GPS receiver units, and could be incorporated for future tests.

5.3 Future Research

Several aspects of the work presented could benefit from future research. As was mentioned, little effort was given to the implementation efficiency of the algorithms developed. To use the techniques presented to incorporate EL into a real-time system, these inefficiencies should be addressed.

More fundamentally, there are also aspects of the algorithms which should be studied in more detail. A data set with ground truth for vehicle trajectories and map landmark locations should be used to evaluate more precisely the accuracy improvements gained with the use of EL using the techniques developed herein.

For the experiments conducted here, a fixed gate was used to qualify the use of EL transformation estimates. This gate was determined empirically for the data set used. A more rigorous methodology to do this could be a subject of future research. A trade-off analysis could be prepared to study the effects of varying these gate parameters on the overall accuracy and performance of the algorithm.

And lastly, as suggested in the previous section, research should be condcuted to incorporate the EL techniques from this thesis with other non-EKF-based SLAM techniques, such as particle filters.

Appendix A

Variation of the Mahalanobis Distance

When determining the distance between points of a multi-dimensional Gaussian distribution, standard Euclidean distance is not a meaningful metric. In statistics, the Mahalnobis distance is a commonly used metric which utilizes the correlations between the variables and is scale-invariant [25]. This measure is widely used in cluster analysis and pattern matching.

The Mahalanobis distance is also used to measure dissimilarity between random vectors of the same distribution. More specifically, given the random variables $\vec{x}$ and $\vec{y}$ from the Guassian distribution with a covariance matrix of Σ, the Mahalanobis distance is defined as:

$$d(\vec{x}, \vec{y}) = \sqrt{(\vec{x} - \vec{y})^T \Sigma^{-1} (\vec{x} - \vec{y})} \tag{A.1}$$

The Euclidian distance is a special case of this distance metric when the covariance matrix is the identity matrix.

In the situation being studied, a metric of distance between points from the local and external frames is required. These points are modeled as variables with random Gaussian distributions, each with a mean and a covariance, representing uncertainty. As such, the Euclidean distance is not sufficient to accurately represent the distance between these points.

The Mahalanobis distance could be used, however, it requires that the points be from

the same distribution. In this fitting process, the points being compared are from completely independent distributions. As a result, a variation of the Mahalanobis distance is used. The covariance matrices are combined, to yield a single distribution for the two points. Since the measurements are assumed to be from independent distributions, their combined distribution is approximated as the sum of their respective covariance matrices. It is assumed that the sensitivity of the distance metric is relatively linear with respect to changes in the covariances, thus they are directly summed.

More specificially, given two measurements from different distributions, $\vec{x_S}$ and $\vec{x_E}$, with the respective uncertainty measures of Σ_S and Σ_E, the distance is measured as:

$$d(\vec{x_S}, \vec{x_E}) = \sqrt{(\vec{x_S} - \vec{x_E})(\Sigma_S + \Sigma_E)^{-1}(\vec{x_S} - \vec{x_E})} \tag{A.2}$$

As will be seen in Section 3.2.2, if these points are in different frames, they must be first transformed to the same measurement frame. The mean can be transformed from **S** to **E** with Equation 3.9 and the covariance transformed using:

$$^{E}\Sigma_s(t) = R(\theta)^{S}\Sigma_s(t)R(\theta)^{T} \tag{A.3}$$

References

[1] Daniel C. Asmar, John S. Zelek, and Samer M. Abdallah. Tree trunks as landmarks for outdoor vision slam. In *CVPRW '06: Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop*, page 196, Washington, DC, USA, 2006. IEEE Computer Society.

[2] N. Ayache and O. Faugeras. Maintaining a representation of the environment of a mobile robot. *IEEE Transactions on Robotics and Automation*, 5(6):804–819, 1989.

[3] T. Bailey. *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, University of Sydney, Sydney, NSW, Australia, 2002.

[4] T.A. Bailey. Constrained initialisation for bearing-only slam. In *2003 IEEE Int. Conf. Robotics and Automation*, pages 1966–1971.

[5] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press, 1988.

[6] Ake Bjorck. *Numerical Methods for Least Squares Problems*. SIAM Press, Philadelphia, PA, 1996.

[7] Michael Carsten Bosse. *ATLAS: A Framework for Large Scale Automated Mapping and Localization*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 2004.

[8] N.A. Carlson. Federated filter for fault-tolerant integrated navigation systems. In *Position Location and Navigation Symposium*, Winchester, MA, 1988.

[9] H Chernoff and E.L Lehmann. The use of maximum likelihood estimates in χ^2 tests for goodness-of-fit. *The Annals of Mathematical Statistics*, 25:579–586, 1954.

[10] P.H. Dana. http://www.colorado.edu/geography/gcraft/notes/gps/gif/ecefxyz.gif, 1994.

[11] C. Estrada, J. Neira, and J.D. Tardos. Hierarchical slam: Real-time accurate mapping of large environments. 21.

[12] Jay Farrell and Matthew Barth. *The Global Positioning System & Inertial Navigation*. McGraw-Hill, New York, NY, USA, 1999.

[13] H.J.S. Feder, J.J. Leonard, and C.M. Smith. Adaptive mobile robot navigation and mapping. 18.

[14] J. Guivant and E. Nebot. Optimization of the simultaneous localization and map building algorithm for real time implementation. 2001.

[15] J. Guivant and E. Nebot. Optimization of the simultaneous localization and map-building algorithm for real-time implementation. *IEEE Transactions on Robotics and Automation*, 17:41–76, 2001.

[16] Jose Guivant and Eduardo Nebot. Simultaneous localization and map building: Test case for outdoor applications. http://www-personal.acfr.usyd.edu.au/nebot/publications/slam/IJRR_slam.htm.

[17] B. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, 4, 1987.

[18] S. Julier and J. Uhlmann. A nondivergent estimation algorithm in the presence of unknown correlations. In *Proceedings of the American Control Conference*.

[19] Simon J. Julier and Jeffery K. Uhlmann. A new extension of the kalman filter to nonlinear systems. In *AeroSense: The 11th International Symposium on Aerospace/Defense Sensing, Simulation and Controls, Multi Sensor Fusion, Tracking and Resource Management II*, 1997.

[20] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. Fast incremental square root information smoothing. pages 2129–2134, Hyderabad; India, 2007.

[21] J. Kim and S. Sukkarieh1. 6dof slam aided gnss/ins navigation in gnss denied and unknown environments.

[22] J. Leonard, H. Durant-Whyte, and I. Cox.

[23] J. J. Leonard and R. J. Rikoski. Incorporation of delayed decision making into stochastic mapping. In *In International Symposium On Experimental Robotics*, 2000.

[24] J.J. Leonard and H.J.S. Feder. A computationally efficient method for large-scale concurrent mapping and localization. In *Proceedings of the Ninth International Symposium on Robotics Research*.

[25] P.C. Mahalanobis. On the generalised distance in statistics. In *Proceedings of the National Institute of Science of India*, volume 12, 1936.

[26] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11, 1963.

[27] A. Martinelli, V. Nguyen, N. Tomatis, and R. Siegwart. A relative map approach to slam based on shift and rotation invariants. 55:50.

[28] M. Montemerlo. *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem with Unknown Data Association*. PhD thesis, Robotics Institute, Carnegie Mellon University, 2003.

[29] Luis Ernesto Navarro-Serment, Chris Paredis, and Pradeep Khosla. A beacon system for the localization of distributed robotic teams. In *Proceedings of the International Conference on Field and Service Robotics (FSR '99)*, August 1999.

[30] A. Nchter, H. Surmann, K. Lingemann, J Hertzberg, and S. Thrun. 6d slam with application in autonomous mine mapping. In *Proceedings IEEE 2004 International Conference Robotics and Automation*, New Orleans, USA, 2004.

[31] J. Nieto, J. Guivant, E. Nebot, and S. Thrun. Real time data association for FastSLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

[32] Roland Siegwart and Illah Nourbakhsh. *Introduction to Autonomous Mobile Robots*. The MIT Press, Boston, Massachusetts, 2004.

[33] R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. *Autonomous Mobile Robots : Perception, Mapping and Navigation*, 1:323–330, 1987.

[34] V.A. Tupysev. A generalized approach to the problem of distributed kalman filtering. In *AIAA Guidance, Navigation and Control Conference*, 1998.

[35] S. Williams. *Efficient Solutions to Autonomous Mapping and Navigation Problems*. PhD thesis, The University of Sydney, 2001.