Visual Trajectory Based SLAM

Isaac Esteban

March 27, 2008

UNIVERSITEIT VAN AMSTERDAM

Master's thesis Artificial Intelligence,
specialization in Intelligent Systems

Under supervision of
Ben Krose,
Olaf Booij
and Zoran Zivkovic
January 2008

by
Isaac Esteban

Intelligent Autonomous Systems Lab Amsterdam
Informatics Institute
Faculty of Science
University of Amsterdam

ISLA

Contents

Chapter 1

Introduction

SLAM stands for Simultaneous Localization And Mapping. It is a fundamental topic in Autonomous Systems and Robotics as it represents one of the most basic skills that any robot requires in order to be truly autonomous. This skill will allow a robot placed in an unknown environment at an unknown location to simultaneously build a consistent spatial representation (a map) and determine its location within this map.

The classical approach to solve the SLAM problem was presented long ago [21] and many alternative solutions have been published since. The objective of any SLAM algorithm is to estimate as accurately as possible both the map and the robot location and their uncertainties based on the information conveyed by one or more sensors. The potential application of SLAM is incredibly wide, ranging from indoor and outdoor robots to underwater and aerial vehicles. Despite the fact that the SLAM problem is considered theoretically solved, many new developments are published every year that aim at the definition of a simple, practical and affordable solution, and try to overcome many of the practical issues that remain unsolved.

The standard approach to solving the SLAM problem is considering the robot pose and the map as a process that needs to be estimated. In such a process, the robot moves within the environment taking measurements that convey certain information that can be related to the estimation of its position within the map. This approach is a direct match to the state estimation methods, where certain process is estimated using information about both the dynamics of the process and the information that is obtained through measurement.

The range of sensors used to measure the environment and convey information about the robot pose and the map, vary as much as the algorithms published during the last years. Laser range scanners are considered one of the most robust measurement devices due to their accuracy, though they present some disadvantages such as the price and limited range of applicability. Most laser range sensors are only suitable for indoor environments and those which are valid for outdoors are extremely expensive.

Computer Vision based approaches have receive a lot ot attention in the recent literature due to the flexibility and affordability of cameras. A lot of the reseach in the SLAM topic is nowadays based on some form of visual SLAM.

1.1 Related Work

S. Pfister [18] approaches the problem of building a map using a robot equipped with laser range scanner by extracting a set of segments and their uncertainty from the raw laser scans data. They present an algorithm that first extracts a simple set of lines using the Hough transform and then merges those lines through a line fitting algorithm. In order to estimate the uncertainty of the obtained lines, error models for the laser scanner are used. Their approach keeps a global map consisting of lines that are re-merged at every robot location where a new laser scan is obtained. The essential disadvantage of this approach is the need to either have an extremely accurate odometry measure to perform a correct line fitting from one robot pose to the next one, or the definition of very complex correlation terms in the error models for the fitted lines. F. Lu and E. Milios [16] present an alternative method in which laser scans are consistently aligned. Based on the maximum likelihood criterion they propose a method in which all the local frames (scans at each robot pose) are kept as well as the relative spatial relationships between them. These spatial relationships are modeled as random variables and derived from either odometry or matching pairwise scans. They achieve consistency by using all the spatial relations as constraints to solve for the data frame poses simultaneously instead of the classical incremental approach.

Despite of the accuracy of laser range scanners, they are still expensive and not practical for many applications. In the recent literature, cheaper and more flexible measurement devices such as cameras have gained a lot of popularity in the SLAM community. Cameras represent a cheap and flexible solution for measuring the environment and they convey information about the map and the relative pose of the robot. They are flexible as many different feature extraction techniques can be applied as they are naturally a content rich sensor. From simple corner detectors to very complex pairwise pose estimation methods with omnidirectional cameras can be used to extract information about the environment. P. Jensfelt et. al. [13] present a method based on the extraction and matching of image features using a single affordable camera. By means of a modified rotationally variant SIFT [15] descriptor they consider only a few high quality image features to represent the map and use in a SLAM algorithm. As a single camera is used only bearing information about the image features is known.

A.J. Davidson et al. [2] also approach the SLAM problem with a single camera. Furthermore they present a *real time* algorithm that can recover the 3D trajectory of a single uncontrolled monocular camera moving rapidly through a previously unknown scene. They apply their methods to a humanoid robot equiped with a wide vision monocular camera moving in small circles. However, they also included the use of a gyro as an additional sensor in the SLAM algorithm. Their approach is based on the on-line creation of a sparse persistent map of natural landmarks within a probabilistic framework.

Alternative approaches to SLAM using cameras also include more complex devices such as stereo vision cameras or omnivision systems. Using an omnidirectional camera, O. Booij et. al. [3] present a method in which by measuring similarity [14] between images taken at every robot pose they build a topological map that is then used for navigation. They achieve robustness by means of the epipolar geometry and a planar floor constraint that allows the computation of the relative heading of pairwise images. By means of image similarity the robot equipped with the camera is capable of recognizing previously seen areas.

As we have seen, the wide range of possible features that can be extracted from any measurement device yields an enormous amount of possibilities with regard to the feature representation (i.e. the map can be 2D landmark locations, bearing, 3D positions, distances, etc). J. Folkesson et. al. [8] propose a feature representation framework that addresses symmetries and constraints in the feature coordinates. Their approach also allows the features to be initialized with partial information only. This is particularly useful when using a single camera when at first only bearing information is available. In this framework the number of dimensions of a feature grows as more information is gathered. Such a generic approach is also suitable for sensor fusion where laser and camera information can be combined.

Approaching the feature or map representation with a different angle, R. Eustice et al. [6] present a method for the estimation of the trajectory of an underwater vehicle using a 5 DOF camera. The robot trajectory is represented by a history of robot poses and overlapping imagery provide partial observation of these poses. Their delayed state representation, also termed trajectory based, is framed in a Kalman Filter state estimator.

1.2 Visual Trajectory Based SLAM

The Kalman Filter [23] (KF) and its variant the Extended Kalman Filter (EKF) are considered the standard solution to the SLAM problem. They represent a recursive solution to the problem of estimating the state of a process in the presence of certain measurements taken from the environment. In our robotics environment, the process to be estimated is the map and the robot location, and the measurements are the features extracted from whichever sensor the robot is equipped with. There are however a number of difficulties that make the application of the KF SLAM algorithms impossible to real life applications. Firstly, for large environments where a large map full of features needs to be created, the computational complexity of most solutions makes the application for mapping large areas unrealistic and not feasible. Secondly, there is a natural problem associated with the event at which the robot revisits previously seen areas, the so called *loop closing* problem.

In this Thesis we present a combined method that aims at the solution of both the computational complexity and the loop closing problems in the SLAM framework. Using a single omnidirectional camera, we combine a robust omnivision image matching algorithm, a relative pose estimation method [4] [14] and a trajectory based state representation [7]. We embed this within an Information Filter [19] to obtain an Omnivision Trajectory Based SLAM algorithm. We show in our experiments that our combined method can accurately build a map of the environment in *almost* linear time in the size of the map and can accurately close the loop even when the accumulated error in the odometry is larger than 80 meters in distance and 100 degrees in orientation. Finally we show that our approach can cope with both the computational complexity and the loop closing problem using a single omnivision camera for extremely large environments.

1.3 Organization of this thesis

The content of this Thesis is organized in 5 chapters. In Chapter 2 we introduce the State Estimation problem based on the Kalman Filter solution and its variants the Extended Kalman Filter and the Extended Information Filter. This chapter represents the theoretical background required to understand our approach and is presented with a generic view with no reference but the essential to our particular implementation. We explore in a linear fashion the probabilistic framework in which we embed our estimation approach, paying special attention to the derivation of the essential formulas in the begining (KF and EKF) which make the understanding of the Information Filter much more accesible. We also present the formal methodology to measure the error introduced with the Extended Kalman Filter by means of consistency measurement. In Chapter 3 we present our approach in detail and discuss both our measurement methodology to obtain information about the environment and the probabilistic framework used to estimate the robot trajectory. We pay particular attention to the definition of the mathematical form of the state representation and give detailed information about the computational benefits of the Information Filter for trajectory based approaches. In Chapter 4 we present our experimental setup as well as the results obtained. We show by means of three different experiments three essential results. We first discuss an artificially created data set and explore how the errors introduced by the EKF affect the trajectory estimation. A data set taken in a small office enviroment is used to illustrate how the loop closure is performed and how accurate our method is in the presence of a sufficiently bright images. Finally, we present the largest data set ever used to our knowledge in a SLAM framework where we show how the loop is effectively closed for an extremely large environment. In Chapter 5 we summarize our approach and results, and draw the final conclusions and discuss future work. Additionally, two Appendixes are provided to present some mathematical background and a summary of all the relevant notation.

Chapter 2

Introduction to State Estimation

The problem of building a map and localizing a robot within the map can be understood as a state estimation problem. The robot moving around the environment and taking measurements represents the process, while the robot location and the map are the *state* of this process that need to be estimated. By means of incorporating information about the environment inferred from a (noisy) sensor device, a state estimation procedure aims at the discovery of the state that best describes the process at any given time by filtering out the noise. In this chapter we present the theoretical background related to our work on the state estimation problem (estimating the map and the relative robot location). We first present the basics of state estimation, namely the Kalman Filter, and then extend its application to non linear processes, the Extended Kalman Filter. Finally, we study an alternative representation of the Extended Kalman Filter which presents computational advantages.

2.1 Kalman Filter

2.1.1 Introduction

The Kalman Filter (KF) is a recursive solution to the problem of estimating the present, past or future state of a process minimizing the mean of the squared error. The KF is used to estimate the state $x \in \mathbb{R}^n$ of a discrete time process that is governed by a *linear* stochastic difference equation. Such process consist of two elements: the control of the process (the dynamics) and the measurements $z \in \mathbb{R}^m$ taken during the process.

Process Control

The control equation defines the state of the process as a linear combination of the state in the previous time step, the control input and some error:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_k, \tag{2.1}$$

where x_k is the state of the process in time step k, $A(n \times n)$ relates the state in time step $k-1$ to the state in time step k (modeling the dynamics), $B(n \times l)$ relates the control input $u \in \mathbb{R}^l$ to the state x_k and $w_k \in \mathbb{R}^n$ is the random variable that represents the process noise at time step k.

As a convention in the SLAM community, the odometry measurement is usually taken as the control input , being the error w_k in that case the error associated with the odometry model. As we followed this convention in our work, the words *control* and *odometry* will be interchangeable and the error w_k will be defined as the error of the odometry model.

Measurements

The measurement equation defines the measurement prediction as a linear combination of the state of the process and the error for the measurement model:

$$z_k = Hx_k + v_k, \tag{2.2}$$

where z_k is the measurement prediction at time step k, $H(m \times n)$ relates the state of the process to the measurement taken at time step k and $v_k \in \mathbb{R}^m$ is the random variable that represents the measurement error.

Gaussian Assumption

One essential aspect of the KF is the assumption that is made over the random variables (the noises) v_k and w_k. They are both assumed to be independent from each other, white, and with normal probability distribution [1]:

$$p(w) \sim \mathcal{N}(0, Q), \tag{2.3}$$

$$p(v) \sim \mathcal{N}(0, R), \tag{2.4}$$

where Q represents the error covariance of the process, and R the error covariance of the measurement.

Origins of the KF

The KF works in two steps that are recursively repeated, the *time update* step and the *measurement update* step. During the time update step a prediction of the state (*a priori* estimation) is performed based entirely on the dynamics of the process. During the measurement update step, this estimation is updated with the information that the measurements convey (*a posteriori* estimation).

[1]Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ with mean μ and covariance matrix Σ

The a priori state estimate is easily derived from equation 2.1 (taking the noise out) and is defined as a linear combination of the previous state and the control input:

$$\hat{x}_k = Ax_{k-1} + Bu_{k-1}. \tag{2.5}$$

The goal of the KF is to compute the *a posteriori* state estimate x_k as a linear combination of an *a priori* estimate and a weighted difference between the real measurement and the measurement prediction.

We will define x_k as the *a posteriori* state estimate at time step k, $\hat{x}_k$ the *a priori* state estimate, $\check{x}_k$ the ground truth state, $\hat{z}_k$ the measurement estimate and z_k the actual measurement taken at time step k (for more details on notation see Appendix B).

Following these definitions, the equation we need to find is:

$$x_k = \hat{x}_k + K(z_k - \hat{z}_k), \tag{2.6}$$

where the difference $z_k - \hat{z}_k$ is the *measurement innovation* and $K(n \times m)$ is the weighting factor. As the measurement prediction $\hat{z}_k$ can be expressed as $H\hat{x}_k$ (see equation 2.2), an equivalent expression is:

$$x_k = \hat{x}_k + K(z_k - H\hat{x}_k), \tag{2.7}$$

The a posteriori state estimate (2.7) represents the mean or first moment of the state distribution [23] and it is normal if the Gaussian assumptions 2.3 and 2.4 are met.

For both the *a priori* and *a posteriori* state estimates there is an *a priori* and *a posteriori* estimate error covariance that represents the accuracy of the estimate:

$$\hat{P}_k = E[\hat{e}_k \hat{e}_k^T], \tag{2.8}$$

$$P_k = E[e_k {e_k}^T], \tag{2.9}$$

where $\hat{e}_k$ is the a priori error $(\check{x}_k - \hat{x}_k)$ and e_k is the a posteriori error $(\check{x}_k - x_k)$.

The a posteriori estimate error covariance (see equation 2.9) represents the variance of the state distribution [23]:

$$p(x_k \mid z_k) \sim \mathcal{N}(\hat{x}_k, P_k). \tag{2.10}$$

In order to optimize the state estimation the weighting factor K (also called gain) is chosen to minimize the a posteriori error covariance (see equation 2.9). After a number of transformations involving equations 2.8, 2.9 and the definition of the error e_k (for more information see [12], [5], [17] and [23]), the expression of the gain is simplified (in one of its common forms) as:

$$K_k = \frac{\hat{P}_k H^T}{\hat{P}_k H^T + R}, \tag{2.11}$$

where R is the measurement error covariance[1]. In other words, looking at expression 2.11, the weighting factor K gives more credit to the measurement innovation as the R diminishes, while it gives less credit to it as the $\hat{P}_k$ diminishes.

Starting with the equation we wanted to derive (see equation 2.6) and the definition of the gain K, we have reached a weighted linear combination of the a priori state estimate and the measurement innovation, to estimate the state of the process minimizing the mean of the squared error or error covariance:

$$x_k = \hat{x}_k + \left(\frac{\hat{P}_k H^T}{\hat{P}_k H^T + R} \right) (z_k - H\hat{x}_k), \tag{2.12}$$

[1] R is usually found in the literature without the time step index k as it is commonly taken independent of the time step as for many models it is difficult to make an online estimation of the measurement error. The same applies for Q, which is in general assumed independent of the time step. We will make the same assumption though all our work.

2.1.2 KF in Probabilistic Terms

Two of the fundamental aspects of the KF are the *Gaussian Assumption* and the *linear* stochastic difference equation that governs the process to be estimated. However, it is difficult to grasp these probabilistic concepts and their consequences from the definition of the equations of the KF. In order to get a clear picture of how the probabilities of both the state x and the measurements z work together and to find a rationale for the equations of the kalman filter algorithm, we will successively review the probability distribution of the state x, probability distribution of the measurement z, the joint probability of x and z, and finally the distribution of the conditional $x \mid z$ and $z \mid x$, which will prove to be essential for the later understanding of the *non linear* version of the KF.

As a tool for the derivation of an expression for these distributions, we will make use of the canonical representation of a Gaussian distribution.

Probability distribution of measurement z

The probability of the measurement z is a Gaussian distribution (see 2.1.1) with mean the predicted measurement $\hat{z}$ and covariance matrix R.

$$P(z) = \mathcal{N}(z; \hat{z}, R). \tag{2.13}$$

Substituting $\hat{z}$ for its equivalent $H\hat{x}$ (see equation 2.2),

$$P(z) = \mathcal{N}(z; H\hat{x}, R), \tag{2.14}$$

and expressing the normal distribution with its common formulation yields:

$$P(z) = \frac{1}{\sqrt{2\pi R}} exp\left(-\frac{1}{2}(z - H\hat{x})^T R^{-1}(z - H\hat{x})\right). \tag{2.15}$$

The quadratic in the exponential can be expanded:

$$P(z) = \frac{1}{\sqrt{2\pi R}} exp\left(-\frac{1}{2}\left(z^T R^{-1} z - 2(H\hat{x})^T R^{-1} z + (H\hat{x})^T R^{-1} H\hat{x}\right)\right), \tag{2.16}$$

and the first part introduced in the exponential:

$$P(z) = exp\left(-\frac{1}{2} log \mid 2\pi R \mid -\frac{1}{2} z^T R^{-1} z + (H\hat{x})^T R^{-1} z - \frac{1}{2}(H\hat{x})^T R^{-1} H\hat{x}\right). \tag{2.17}$$

Looking closely to the above expression and reorganizing the terms:

$$P(z) = exp\left(\left(-\frac{1}{2} log \mid 2\pi R \mid -\frac{1}{2}(H\hat{x})^T R^{-1} H\hat{x}\right) + \left((H\hat{x})^T R^{-1} z\right) - \left(\frac{1}{2} z^T R^{-1} z\right)\right), \tag{2.18}$$

the above equation is precisely the canonical representation (also called *information form*) of a Gaussian (see Appendix A.1):

$$P(z) = exp(g + m^T z - \frac{1}{2} z^T W z), \tag{2.19}$$

where

$$\begin{aligned} m &= R^{-1} H \hat{x}, & (2.20)\\ W &= R^{-1}, & (2.21)\\ g &= -\frac{1}{2} log \mid 2\pi R \mid -\frac{1}{2} H \hat{x}^T R^{-1} H \hat{x}. & (2.22) \end{aligned}$$

Hence, the *information form* of the Gaussian distribution of measurement z can be expressed as:

$$P(z) = \mathcal{N}^{-1}(z; R^{-1} H \hat{x}, R^{-1}). \tag{2.23}$$

Probability distribution of state x

Following the same procedure as with the measurement, the probability of state x is defined as a Gaussian distribution with mean the a priori state estimate and covariance matrix P, which is estimated in the KF algorithm:

$$P(x) = \mathcal{N}(x; \hat{x}, P), \tag{2.24}$$

which can be expressed in its canonical form as:

$$P(x) = \mathcal{N}^{-1}\left(x; P^{-1}\hat{x}, P^{-1}\right). \tag{2.25}$$

Joint distribution for measurement and state

We are now interested in the joint distribution of the measurement z and the state x (see Appendix A.2). For these two variables we know from equation 2.2:

$$z = Hx + v, \tag{2.26}$$

where $p(v) \sim \mathcal{N}(0, R)$ and $p(x) \sim \mathcal{N}(\hat{x}, P)$, hence, making use of Bayes rule, the joint distribution is characterized as :

$$P(x, z) = \mathcal{N}\left(\begin{bmatrix} \hat{x} \\ H\hat{x} \end{bmatrix}, \begin{bmatrix} P & PH^T \\ HP & HPH^T + R \end{bmatrix} \right). \tag{2.27}$$

To proceed we need to compute the inverse of the covariance matrix Σ of $P(x,z)$ in order to reach a canonical representation of the joint distribution.

$$\Sigma^{-1} = K = \begin{bmatrix} K_{xx} & K_{xz} \\ K_{zx} & K_{zz} \end{bmatrix}. \tag{2.28}$$

Such expression of the inverse can be found in Appendix A.2 in the Inversion Lemma, which after a number of simplifications leads to:

$$K = \begin{bmatrix} P^{-1} + H^T R^{-1} H & -H^T R^{-1} \\ -R^{-1} H & R^{-1} \end{bmatrix}. \tag{2.29}$$

So the joint Gaussian distribution of the measurement z and the state x can be now characterized using the canonical representation as:

$$P(x,z) = exp\left(g + m^T \begin{bmatrix} x \\ z \end{bmatrix} - \frac{1}{2} \begin{bmatrix} x & z \end{bmatrix} W \begin{bmatrix} x \\ z \end{bmatrix} \right), \tag{2.30}$$

where:

$$\begin{aligned} m &= \begin{bmatrix} P^{-1}\hat{x} \\ 0 \end{bmatrix}, & (2.31) \\ W &= \begin{bmatrix} P^{-1} + H^T R^{-1} H & -H^T R^{-1} \\ -R^{-1} H & R^{-1} \end{bmatrix}, & (2.32) \\ g &= -\frac{1}{2} log \mid 2\pi P \mid - \frac{1}{2} \hat{x}^T P^{-1} \hat{x} - \frac{1}{2} log \mid 2\pi R \mid . & (2.33) \end{aligned}$$

Or using the standard notation:

$$P(x,z) = \mathcal{N}^{-1}\left(x, z; \begin{bmatrix} P^{-1}\hat{x} \\ 0 \end{bmatrix}, \begin{bmatrix} P^{-1} + H^T R^{-1} H & -H^T R^{-1} \\ -R^{-1} H & R^{-1} \end{bmatrix} \right). \tag{2.34}$$

Having obtained expressions for the distribution of z, x and the joint distribution of both z and x, we now have the basic tools to derive the expression of the a posteriori error covariance along with some other useful expressions.

Conditional distribution of the measurement given the state

From Bayes Rule we know that:

$$P(z \mid x) = \frac{P(x,z)}{P(x)} \tag{2.35}$$

We already have an expression for both distributions in the right hand side. It is easy to show that the above joint distribution $P(x,z)$ can be expressed as a multiplication of two distributions in its canonical form (it is easy to show how it works as the contents of the exponentials can be added to represent a multiplication). The joint distribution for state x and measurement z (see equation 2.34) can be expressed as the multiplication of two distributions:

$$exp\left(g_1 + \begin{bmatrix} P^{-1}\hat{x} \\ 0 \end{bmatrix}^T \begin{bmatrix} x \\ z \end{bmatrix} - \frac{1}{2}\begin{bmatrix} x & z \end{bmatrix}\begin{bmatrix} P^{-1} & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ z \end{bmatrix}\right) \times$$
$$exp\left(g_2 + \begin{bmatrix} 0 \\ 0 \end{bmatrix}^T \begin{bmatrix} x \\ z \end{bmatrix} - \frac{1}{2}\begin{bmatrix} x & z \end{bmatrix}\begin{bmatrix} H^TR^{-1}H & -H^TR^{-1} \\ -R^{-1}H & R^{-1} \end{bmatrix}\begin{bmatrix} x \\ z \end{bmatrix}\right), \tag{2.36}$$

where:

$$g_1 = -\frac{1}{2}log \mid 2\pi P \mid -\frac{1}{2}\hat{x}^TP^{-1}\hat{x}, \tag{2.37}$$
$$g_2 = -\frac{1}{2}log \mid 2\pi R \mid \tag{2.38}$$

The first term in equation 2.36 is the canonical representation for the distribution of x (see equation 2.25), hence the second term, following the Bayes equation 2.35 must be the canonical representation of $P(z \mid x)$.

$$P(z \mid x) = \mathcal{N}^{-1}\left(z, x; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} H^TR^{-1}H & -H^TR^{-1} \\ -R^{-1}H & R^{-1} \end{bmatrix}\right) \tag{2.39}$$

Conditional distribution of a particular measurement given the state

We now must introduce in the conditional distribution one particular given measurement (the one taken at the interest time step k) so that the probability of the measurement z to be the actual measurement z^R ($P(z = z^R \mid x)$) can be calculated. This is rather simple as the canonical representation for such distribution will be simplified as z is no longer a variable, but a vector. Lets begin with the expression for the conditional distribution $P(z \mid x)$ as show in equation 2.39:

$$exp\left(-\frac{1}{2}log \mid 2\pi R \mid + \begin{bmatrix} 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ z \end{bmatrix} - \frac{1}{2}\begin{bmatrix} x & z \end{bmatrix}\begin{bmatrix} H^TR^{-1}H & -H^TR^{-1} \\ -R^{-1}H & R^{-1} \end{bmatrix}\begin{bmatrix} x \\ z \end{bmatrix}\right) \tag{2.40}$$

As we are looking for the probability of the measurement being z^R, we can rewrite the expression as:

$$exp\left(-\frac{1}{2}log \mid 2\pi R \mid + \begin{bmatrix} 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ z^R \end{bmatrix} - \frac{1}{2}\begin{bmatrix} x & z^R \end{bmatrix}\begin{bmatrix} H^TR^{-1}H & -H^TR^{-1} \\ -R^{-1}H & R^{-1} \end{bmatrix}\begin{bmatrix} x \\ z^R \end{bmatrix}\right) \tag{2.41}$$

and realize the products:

$$exp\left(-\frac{1}{2}log \mid 2\pi R \mid -\frac{1}{2}\left(x^T H^T R^{-1} Hx - z^{R^T} R^{-1} Hx - x^T H^T R^{-1} z^R + z^{R^T} R^{-1} z^R\right)\right) \tag{2.42}$$

and finally reorganize the terms noting that z^R is no longer a variable so we can reach a new canonical representation for the Gaussian conditional distribution for a given measurement:

$$exp\left(\left(-\frac{1}{2}log \mid 2\pi R \mid -\frac{1}{2}z^{R^T} R^{-1} z^R\right) + \left(z^{R^T} R^{-1} Hx\right) - \left(\frac{1}{2}x^T H^T R^{-1} Hx\right)\right) \tag{2.43}$$

which precisely takes the canonical form of a Gaussian distribution:

$$P(z = z^R \mid x) = \mathcal{N}^{-1}\left(z = z^R \mid x; [z^{R^T} R^{-1} H]^T, H^T R^{-1} H\right), \tag{2.44}$$

with the *omitted* constant g:

$$g = -\frac{1}{2}log \mid 2\pi R \mid -\frac{1}{2}z^{R^T} R^{-1} z^R; \tag{2.45}$$

Conditional distribution of the state given one particular measurement

We know that given the covariance and mean of the jointly Gaussians x and z (see equation 2.27), we can compute the conditional distribution, using Bayes rule, of x given z as:

$$P(x \mid z = z^R) = \mathcal{N}\left(x \mid z = z^R; \hat{x} + \left(\frac{PH^T}{HPH^T + R}\right)(z^R - H\hat{x}), P - \left(\frac{PH^T}{HPH^T + R}\right) HP\right), \tag{2.46}$$

where we can see the precise definition of the gain K:

$$K = \frac{PH^T}{HPH^T + R} \tag{2.47}$$

and after some transformations of the covariance expression, the common form of the *a posteriori* covariance estimate:

$$P - \left(\frac{PH^T}{HPH^T + R}\right) HP = (I - KH)P \tag{2.48}$$

This is the expression for the a posteriori error covariance that completes the definition of the KF algorithm.

2.1.3 The Algorithm

We have obtained an a priori approximation for the state (see equation 2.5) and derived an expression for the a posteriori state estimate that minimizes the a posteriori error covariance. We can now define the full KF algorithm as a two stage algorithm: time update equations (*prediction* stage or a priori estimation) and measurement update equations (*update* stage or a posteriori estimation). We have studied the probabilistic details of the Kalman Filter so we can now construct a full algorithm to perform both the a priori and a posteriori state and error covariance estimates:

Time Update equations

$$\hat{x}_{k+1} = Ax_k + Bu_k \tag{2.49}$$
$$\hat{P}_{k+1} = AP_kA^T + Q \tag{2.50}$$

Measurement Update equations

$$K_{k+1} = \frac{\hat{P}_{k+1}H^T}{H\hat{P}_{k+1}H^T + R} \tag{2.51}$$
$$x_{k+1} = \hat{x}_{k+1} + K_{k+1}(z_{k+1} - H\hat{x}_{k+1}) \tag{2.52}$$
$$P_{k+1} = (I - K_{k+1}H)\hat{P}_k \tag{2.53}$$

The recursive nature of the algorithm is one of the most interesting aspects of the filter as it allows an implementation where the process (typically some robot state) is re-estimated at every step when more information (the robot sensing its environment) is available.

It is difficult, given the set of equations that complete the algorithm, to get a clear idea of its computational complexity. Some factors, such as the state to be estimated, are relevant for the estimation of the complexity. Generally, the state will grow as more information is gathered through the measurements. As the state x and the error covariance P get bigger with every time step, the computation time used by the algorithm also grows with time. In the computation of the gain K the inverse of $H\hat{P}_{k+1}H^T + R$ needs to be computed. This inversion is the most expensive computation in the algorithm, so the overall complexity is ruled by the quadratic complexity of this operation.

2.2 Extended Kalman Filter

2.2.1 Introduction

As discussed in the previous section (see 2.1), the KF addresses the problem of estimating the state of a process that is governed by a *linear* difference equation. There are however many scenarios, specially in robotics, where either the process or the relation between the state and the measurement do not follow a linear model [23]. For such situations a KF that linearizes the process is required and is usually referred as Extended Kalman Filter (EKF).

Process Control

The process for which the state we wish to estimate can be governed by a non linear difference equation, hence instead of having a linear combination of the previous state, the control and the error, there is a non linear function f that takes as input the previous state, the control and the error:

$$x_k = f(x_{k-1}, u_{k-1}, w_{k-1}). \tag{2.54}$$

Such a function is typically present when the dynamics of the system include, for instance, non linear acceleration. Given the characteristics of our robotic environment, the control equation **will remain linear** as we will not model either speed or acceleration and will only account for the odometry as the measure for the robot dynamics, thus no linearization is required. Hence, the process control equation remains linear:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_k. \tag{2.55}$$

Measurements

The relation between the state and the measurement can also be governed by a non linear difference equation. It is common to find a non linear function h that takes both the state of the process and the noise as arguments [23]. We will however, for the sake of simplicity, define the measurement as a linear combination of a non linear function of the state and the noise.

$$z_k = h(x_k) + v_k. \tag{2.56}$$

It is essential to note that the basic difference with the linear KF is that the random variable x is no longer Gaussian after the nonlinear transformation ($h(x)$), hence the measurement z will not remain Gaussian as in the linear KF.

Gaussian Assumption

The goal of the EKF is to linearize the non linear difference equation that governs the measurement process in order to use the standard KF procedure. As the KF requires all distributions to be Gaussian, the Gaussian assumption about the random variables describing the noise that was made before, stands now aswell:

$$p(w) \sim \mathcal{N}(0, Q), \tag{2.57}$$

$$p(v) \sim \mathcal{N}(0, R), \tag{2.58}$$

Linearization

The EKF follows the same two stage structure as the KF. In the *time update* step, an a priori state and covariance error are estimated. In the *measurement update* step, the information conveyed by the measurements is used to improve an *a posteriori* estimation.

In the general case, the process control equation might be governed by a non linear function. However, as in our simplified environment where only linear models are present for the dynamics of the process, only the measurement needs to be approximated (linearized).

In the standard KF, the measurement prediction z_k was unique (no a priori nor a posteriori estimate) and defined by $H\hat{x}_k$. As the measurements now do not follow a linear equation and an approximation needs to be performed (linearization), the measurment estimation works, as the state estimation, in two stages.

The a priori measurement estimation (mean measurement) is easily computed by truncating the error v_k in the measurement difference equation and evaluating the function h on the a priori state estimate:

$$\hat{z}_k \approx h(\hat{x}_k). \tag{2.59}$$

Once the a priori estimate can be computed, the a posteriori estimate is approximated (linearized) as a first order approximation based on the Jacobian evaluated at the mean state (the a priori state estimate):

$$z_k \approx \hat{z}_k + J_H(x_k - \hat{x}_k) + w_k, \tag{2.60}$$

where $\hat{z}_k$ is the a priori measurement estimate and J_H is the Jacobian [1] matrix of the function h evaluated at the mean state.

Despite the fact that the a posteriori measurement estimate is not required for the EKF algorithm, it will be essential for the understanding of the probabilities behind the filter.

[1]The Jacobian matrix of partial derivatives is defined as:

$$J_{H\,[i,j]} = \frac{\delta h_{[i]}}{\delta h_{[j]}}(\hat{x}_k). \tag{2.61}$$

2.2.2 EKF in Probabilistic Terms

As with the KF, the probabilistic terms of the inner workings of the filter are lost in the definitions of the algorithm. It is particularly interesting to take a deep look at the probabilities that play a role in the state estimation as the idea of linearization is clearly understood in its probabilistic aspect.

We will only discuss the probabilities that are relevant in the linearization process of the measurement estimate. Those probabilities that are not discussed here, are exactly the same as in the linear KF.

Probability distribution of measurement z

The probability of the measurement z does not follow a Gaussian distribution as z is defined by a non linear stochastic difference equation. However, as we already have an expression for the linearized approximation of the mean of the measurement (see equation 2.59), the distribution of z can be described as:

$$P(z) = \frac{1}{\sqrt{2\pi R}} exp\left(-\frac{1}{2}(z - h(\hat{x}))^T R^{-1}(z - h(\hat{x}))\right). \tag{2.62}$$

Note that the *only* difference with respect to the linear KF is that the estimation of z is now a function $h(\hat{x})$ instead of a matrix multiplication.

Following the same steps as in the linear case (expanding the quadratic, inserting everything in the exponential and reorganizing), a canonical representation can be reached:

$$P(z) = exp\left(\left(-\frac{1}{2}log \mid 2\pi R \mid -\frac{1}{2}h(\hat{x})^T R^{-1} h(\hat{x})\right) + \left(h(\hat{x})^T R^{-1} z\right) - \left(\frac{1}{2} z^T R^{-1} z\right)\right). \tag{2.63}$$

This expression is a Gaussian distribution (due to the linearization), hence can be expressed in its standard notation:

$$P(z) \approx exp(g + m^T z - \frac{1}{2} z^T W z), \tag{2.64}$$

where

$$\begin{aligned} m &= R^{-1}(h(\hat{x})), & (2.65)\\ W &= R^{-1}, & (2.66)\\ g &= -\frac{1}{2}log \mid 2\pi R \mid -\frac{1}{2}h(\hat{x})^T R^{-1} h(\hat{x}). & (2.67) \end{aligned}$$

Or equivalently:

$$P(z) \approx \mathcal{N}^{-1}(z; R^{-1}h(\hat{x}), R^{-1}). \tag{2.68}$$

Joint distribution for measurement and state

Again, following the same idea as in the linear KF and having the relation:

$$z \approx h(\hat{x}) - J_H(x - \hat{x}) + v, \tag{2.69}$$

defined as the linear approximation of the measurement estimation, we can define the joint distribution of the state x and the approximation of the measurement z as a Gaussian distribution characterized as:

$$P(x,z) \approx \mathcal{N}\left(\left[\begin{array}{c} \hat{x} \\ h(\hat{x}) \end{array}\right], \left[\begin{array}{cc} P & P{J_H}^T \\ J_H P & J_H P {J_H}^T + R \end{array}\right]\right). \tag{2.70}$$

Inverting the covariance matrix of $P(x,z)$ (see equation), again by means of the inversion lemma followed by some simplifications, in order to reach a canonical representation of the joint distribution, the joint Gaussian distribution of the first order approximation of measurement z and the state x can be now characterized using the canonical representation as:

$$P(x,z) \approx exp\left(g + m^T \left[\begin{array}{c} x \\ z \end{array}\right] - \frac{1}{2}\left[\begin{array}{cc} x & z \end{array}\right] W \left[\begin{array}{c} x \\ z \end{array}\right]\right), \tag{2.71}$$

where:

$$\begin{aligned}
m &= \left[\begin{array}{c} P^{-1}\hat{x} + {J_H}^T R^{-1} J_H \hat{x} - {J_H}^T R^{-1} h(\hat{x}) \\ -R^{-1} J_H \hat{x} + R^{-1} h(\hat{x}) \end{array}\right], && (2.72) \\
W &= \left[\begin{array}{cc} P^{-1} + {J_H}^T R^{-1} J_H & -{J_H}^T R^{-1} \\ -R^{-1} J_H & R^{-1} \end{array}\right], && (2.73) \\
g &= -\frac{1}{2} log \mid 2\pi P \mid && (2.74) \\
&\quad -\frac{1}{2} log \mid 2\pi R \mid \\
&\quad -\frac{1}{2} \hat{x} P^{-1} \hat{x} \\
&\quad -\frac{1}{2} \hat{x} {J_H}^T R^{-1} J_H \hat{x} \\
&\quad +h(\hat{x}) R^{-1} J_H \hat{x} \\
&\quad -\frac{1}{2} h(\hat{x}) R^{-1} h(\hat{x})
\end{aligned}$$

Or using the standard notation:

$$P(x,z) \approx \mathcal{N}^{-1}\left(x,z; \begin{bmatrix} P^{-1}\hat{x} + {J_H}^T R^{-1} J_H \hat{x} - {J_H}^T R^{-1} h(\hat{x}) \\ -R^{-1} J_H \hat{x} + R^{-1} h(\hat{x}) \end{bmatrix}, \begin{bmatrix} P^{-1} + {J_H}^T R^{-1} J_H & -{J_H}^T R^{-1} \\ -R^{-1} J_H & R^{-1} \end{bmatrix}\right). \tag{2.75}$$

Conditional distribution of the measurement given the state

Using Bayes Rule we can separate the Gaussian distribution $P(x,z)$ in its canonical form as a product of two Gaussian distributions:

$$exp\left(g_1 + \begin{bmatrix} P^{-1}\hat{x} \\ 0 \end{bmatrix}^T \begin{bmatrix} x \\ z \end{bmatrix} - \frac{1}{2}\begin{bmatrix} x & z \end{bmatrix} \begin{bmatrix} P^{-1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix}\right) \times$$
$$exp\left(g_2 + \begin{bmatrix} {J_H}^T R^{-1} J_H \hat{x} \\ -R^{-1} J_H h(\hat{x}) + R^{-1} h(\hat{x}) \end{bmatrix}^T \begin{bmatrix} x \\ z \end{bmatrix} - \frac{1}{2}\begin{bmatrix} x \\ z \end{bmatrix}^T \begin{bmatrix} {J_H}^T R^{-1} J_H & -{J_H}^T R^{-1} \\ -R^{-1} J_H & R^{-1} \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix}\right), \tag{2.76}$$

where:

$$\begin{aligned} g_1 &= -\frac{1}{2} log \mid 2\pi P \mid -\frac{1}{2}\hat{x}^T P^{-1}\hat{x}, && (2.77) \\ g_2 &= -\frac{1}{2} log \mid 2\pi R \mid && (2.78) \\ & -\frac{1}{2}\hat{x} {J_H}^T R^{-1} {J_H}^T \hat{x} \\ & +\frac{1}{2} h(\hat{x}) R^{-1} {J_H}^T \hat{x} \\ & +\frac{1}{2}\hat{x} {J_H}^T R^{-1} h(\hat{x}) \\ & -\frac{1}{2} h(\hat{x}) P^{-1} h(\hat{x}) \end{aligned}$$

The first term in equation 2.76 is the canonical representation for the distribution $P(x)$ (see equation 2.25), hence the second term, following the Bayes equation 2.35 must be the canonical representation of $P(z \mid x)$.

$$P(z \mid x) \approx \mathcal{N}^{-1}\left(z,x; \begin{bmatrix} {J_H}^T R^{-1} J_H \hat{x} \\ -R^{-1} J_H h(\hat{x}) + R^{-1} h(\hat{x}) \end{bmatrix}, \begin{bmatrix} {J_H}^T R^{-1} J_H & -{J_H}^T R^{-1} \\ -R^{-1} J_H & R^{-1} \end{bmatrix}\right) \tag{2.79}$$

Conditional distribution of a particular measurement given the state

Substituting the stochastic variable z for a particular measurement z^R in equation 2.79 we obtain:

$$exp\left(g_2 + \left[\begin{array}{c} {J_H}^T R^{-1} J_H \hat{x} \\ -R^{-1} J_H h(\hat{x}) + R^{-1} h(\hat{x}) \end{array}\right]^T \left[\begin{array}{c} x \\ z^R \end{array}\right] - \frac{1}{2} \left[\begin{array}{cc} x & z^R \end{array}\right] \left[\begin{array}{cc} {J_H}^T R^{-1} J_H & -{J_H}^T R^{-1} \\ -R^{-1} J_H & R^{-1} \end{array}\right] \left[\begin{array}{c} x \\ z^R \end{array}\right]\right). \tag{2.80}$$

Realizing the products and reorganizing the terms we obtain a canonical representation of the Gaussian approximation (linearized distribution) of $P(z = z^R \mid x)$:

$$P(z = z^R \mid x) \approx exp\left(g + m^T \left[\begin{array}{c} x \\ z^R \end{array}\right] - \frac{1}{2} \left[\begin{array}{cc} x & z^R \end{array}\right] W \left[\begin{array}{c} x \\ z^R \end{array}\right]\right), \tag{2.81}$$

where:

$$\begin{aligned}
m &= {J_H}^T R^{-1} J_H \hat{x} + {z^R}^T R^{-1} J_H, && (2.82)\\
W &= {J_H}^T R^{-1} J_H, && (2.83)\\
g &= -\frac{1}{2} log \mid 2\pi R \mid && (2.84)\\
&\quad -\frac{1}{2} \hat{x} {J_H}^T R^{-1} {J_H}^T \hat{x}\\
&\quad +\frac{1}{2} h(\hat{x}) R^{-1} {J_H}^T \hat{x}\\
&\quad +\frac{1}{2} \hat{x} {J_H}^T R^{-1} h(\hat{x})\\
&\quad -\frac{1}{2} h(\hat{x}) P^{-1} h(\hat{x})\\
&\quad -R^{-1} J_H h(\hat{x}) z^R\\
&\quad +R^{-1} h(\hat{x}) z^R\\
&\quad -\frac{1}{2} {z^R}^T R^{-1} z^R
\end{aligned}$$

Conditional distribution of the state given one particular measurement

Given the covariance and mean of the jointly Gaussians state x and the first order approximation of z (see equation **??**), we can compute the conditional distribution of x given the real measurement z^R as:

$$P(x \mid z = z^R) \approx \mathcal{N}\left(x \mid z = z^R; \hat{x} + \left(\frac{P {J_H}^T}{J_H P {J_H}^T + R}\right)(z^R - h(\hat{x})), P - \left(\frac{P {J_H}^T}{J_H P {J_H}^T + R}\right) J_H P\right), \tag{2.85}$$

where we can see the precise definition of the gain K:

$$K = \frac{P{J_H}^T}{J_H P {J_H}^T + R} \tag{2.86}$$

and after some transformations of the covariance expression, the common form of the *a posteriori* covariance approximated estimate:

$$P - \left(\frac{P{J_H}^T}{J_H P {J_H}^T + R} \right) J_H P = (I - K J_H) P \tag{2.87}$$

Understanding the Linearization

As any approximation method, the linearization performed over the measurement estimation z induces errors, which are transfered to the Gaussian that is fitted to the true distribution.

Lets consider a robotic process where the measurements consist of the distance from the current robot position to the previous known robot position, and the state consist of the collection of robot poses (X,Y and heading α) along the robots trajectory. In such a scenario, the transformation from state to measurement is non linear:

$$h(x_k) = \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2}. \tag{2.88}$$

In this case, the probability of the measurement to be the actual one given the state estimate ($P(z = z^R \mid x)$) will take the shape of a volcano crater centered around the last robot position:

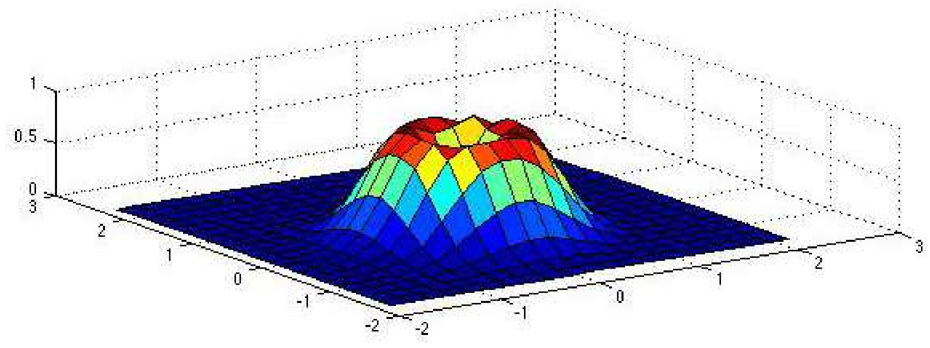

Figure 2.1: True distribution of $P(z = z^R \mid x)$ represented in 3D. The height represents the probability and the coordinates in the plane represent the real world plane coordinates.

In the linearization process, a Gaussian is used to approximate the true distribution. As we can see in the picture below, the particular shape of the true distribution cannot be sufficiently described, hence the error in the estimation of the state and the covariance will be considerable.

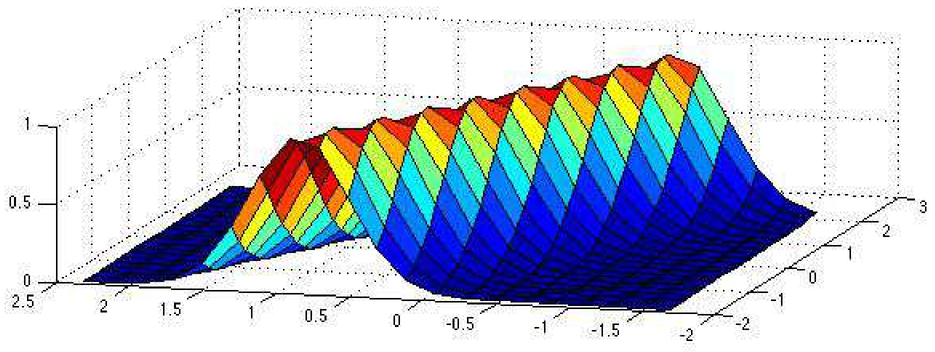

Figure 2.2: Linearization of the distribution of $P(z = z^R \mid x)$ represented in 3D. The height represents the probability and the coordinates in the plane represent the real robot coordinates.

This is perhaps one of the major drawbacks of the EKF and the reason for the filter to become either pessimistic or optimistic. A pessimistic filter estimates the error covariance bigger than the true error covariance. An optimistic filter estimates the error covariance smaller than the true error covariance. Both cases make a filter inconsistent. Further in this section we will discuss some methods to test the filter consistency when ground truth data about the state is known (for instance in simulation) .

2.2.3 The Algorithm

Following the same procedure as with the KF, we can now define a set of equations for the two stages of the EKF: time update equations (*prediction* stage) and measurement update equations (*update* stage).

Time Update equations

$$\hat{x}_{k+1} = Ax_k + Bu_k \tag{2.89}$$

$$\hat{P}_{k+1} = AP_kA^T + Q \tag{2.90}$$

Measurement Update equations

$$K_{k+1} = \frac{\hat{P}_{k+1}J_H^T}{J_H\hat{P}_{k+1}J_H^T + R} \tag{2.91}$$

$$x_{k+1} = \hat{x}_{k+1} + K_{k+1}(z_{k+1} - h(\hat{x}_{k+1})) \tag{2.92}$$

$$P_{k+1} = (I - K_{k+1}J_H)\hat{P}_k \tag{2.93}$$

Note that the essential difference between the KF and the EKF is the introduction of the Jacobian J_H and the function h evaluated at the a priori state estimate that is used to approximate the a priori measurement

2.3 Extended Information Filter

2.3.1 Introduction

One of the principal drawbacks of the EKF is its computation costs. There is a well known barrier [19] around the hundred state elements where the EKF approach becomes no longer viable due to the large execution time. One of the focuses of research in the SLAM community is the reduction of the complexity of the SLAM problem, either through the definition of alternative methods or the use of approximation techniques.

The Information Filter (IF) [19] [7] is an equivalent definition of the KF based on an alternative representation of the Gaussian distribution. For the understanding of the EKF in its probabilistic terms we made use of the information form of the Gaussian distribution, though the filter equations and the estimation process were entirely based on the standard Gaussian representation.

The EIF, on the other hand, is based on the information form representation. If in the EKF the state and error covariance are estimated (a priori and a posteriori), the alternative information vector and information matrix will be estimated in the EIF. It is important to understand that the IF is mathematically equivalent to the KF, though the use of a different representation of the Gaussian distribution yields important advantages in terms of computational costs.

We have already discussed the Extended version of the KF (EKF) hence we will continue on the same line and present the information filter in its extended version, the Extended Information Filter.

2.3.2 Deriving the EIF

We know that the goal of the EIF is to estimate both a priori and a posteriori the information vector and information matrix (also called precision matrix). We also know the formal relation between the information vector η and the estimated state vector x and the information matrix Λ and the error covariance matrix P (see Appendix A.1):

$$\Lambda = P^{-1} \tag{2.94}$$
$$\eta = P^{-1}x \tag{2.95}$$
$$\tag{2.96}$$

With this information and the already defined EKF algorithm, we will now derive the expression for the a priori information matrix estimate, the a posteriori information matrix estimate, the a priori information vector estimate and the a posteriori information vector estimate. All those constitute the EIF algorithm.

Measurement Update

Taking the EKF a posteriori error covariance estimate (see equation 2.70) and including the definition of the Gain, we have:

$$P_{k+1} = (I - \frac{\hat{P}_{k+1} J_H^T}{J_H \hat{P}_{k+1} J_H^T + R} J_H) \hat{P}_{k+1}, \tag{2.97}$$

and multiplying by the a priori error covariance $\hat{P}_k$:

$$P_{k+1} = \hat{P}_{k+1} - \frac{\hat{P}_{k+1} J_H^T}{J_H \hat{P}_{k+1} J_H^T + R} J_H \hat{P}_{k+1}, \tag{2.98}$$

or equivalently:

$$P_{k+1} = \hat{P}_{k+1} - \hat{P}_{k+1} J_H^T [J_H \hat{P}_{k+1} J_H^T + R]^{-1} J_H \hat{P}_{k+1}. \tag{2.99}$$

Expression 2.99 is precisely the definition of **The Matrix Inversion Lemma** (see Appendix A.3), hence can be written as:

$$P_{k+1} = [\hat{P}_{k+1}^{-1} + J_H^T R^{-1} J_H]^{-1}. \tag{2.100}$$

Using the inverted expression, we obtain an expression for the inverse of the a posteriori error covariance which is precisely the a posteriori information matrix:

$$\Lambda_{k+1} = P_{k+1}^{-1} = \hat{P}_{k+1}^{-1} + J_H^T R^{-1} J_H. \tag{2.101}$$

In the above expression $\hat{P}_{k+1}^{-1}$ is precisely the a priori information matrix $\hat{\Lambda}_{k+1}$ which we will derive in the next section:

$$\Lambda_{k+1} = \hat{\Lambda}_{k+1} + J_H^T R^{-1} J_H. \tag{2.102}$$

Time Update

For the time update step the a priori information matrix is defined as the inverse of the a priori error covariance:

$$\hat{\Lambda}_{k+1} = \hat{P}_{k+1}^{-1}. \tag{2.103}$$

Using the expression from the EKF algorithm, we need to obtain the inverse:

$$\hat{P}_{k+1}^{-1} = (AP_kA^T + Q)^{-1}. \tag{2.104}$$

Using the inversion lemma, this is equivalent to:

$$\hat{P}_{k+1}^{-1} = Q^{-1} - Q^{-1}A(A^TQ^{-1}A + P_k^{-1})^{-1}A^TQ^{-1}, \tag{2.105}$$

where P_k^{-1} is the a posteriori information matrix defined in the previous section, hence the a priori information matrix is:

$$\hat{\Lambda}_{k+1} = Q^{-1} - Q^{-1}A(A^TQ^{-1}A + \Lambda_k)^{-1}A^TQ^{-1}. \tag{2.106}$$

2.3.3 The Algorithm

Based on the alternative information form of the EKF, we have derived the expression for both the a priori and a posteriori information matrices. Based on this alternative representation we can now define the complete algorithm in the information form for the EIF.

Time Update equations

$$\hat{\Lambda}_{k+1} = Q^{-1} - Q^{-1}A(A^TQ^{-1}A + \Lambda_k)^{-1}A^TQ^{-1} \tag{2.107}$$

$$\hat{\eta}_{k+1} = \hat{\Lambda}_{k+1}\hat{x}_{k+1} \tag{2.108}$$

Measurement Update equations

$$\Lambda_{k+1} = \hat{\Lambda}_{k+1} + J_H^T R^{-1} J_H \tag{2.109}$$

$$\eta_{k+1} = \Lambda_{k+1}x_{k+1} \tag{2.110}$$

It is interesting to note that the a posteriori information matrix is now much simpler to calculate than the error covariance matrix was. A simple additive operation can now be used instead of a complex inversion. On the other hand, the expression of the a priori information matrix is somehow more complex that its counterpart the a priori error covariance.

2.3.4 State Recovery

The drawback of the EIF is the fact that we no longer obtain an estimation of the state but we do obtain its *relative* in the information form. Once the information matrix and information vector are obtained we need to transform the information vector back to the original mean form in order proceed to the next time step.

The most naive recovery method using the relation between information matrix and covariance matrix results in cubic complexity and voids the efficiency obtained with the new information filter. In fact, recovering the state mean can be described as solving a sparse, symmetric, positive-definite, linear system:

$$\Lambda_t x_k = \eta_t, \tag{2.111}$$

which can be generally solved using the conjugate gradient method (CG, [20]) in n iterations with $O(n)$ complexity per iteration ($O(n^2)$ total cost) and usually in less if a good initialization is used.

In our Matlab experiments we tested five different state recovery techniques. All these techniques are generic solutions to solving the set of equations defined by $Ax = b$.

- **Inversion**: just to use as baseline for improvement measurement, we use the naive inversion technique which uses Gaussian elimination with partial pivoting. This is far from optimal and will make the experiments on large datasets impossible.
- **CGS** (conjugate gradients squared method): this is an iterative solution to the set of equations. However, it requires the definition of an initial guess (in our case we used the state vector of the previous time step augmented with a new robot pose) and a tolerance value of $1e-7$. The solution is not guaranteed to be exact.
- **PCG** (preconditioned conjugate gradients method): if the information matrix is ill conditioned, a preconditioner can be used to speed up the process. For our experiments we used the incomplete Cholesky factorization of the information matrix as preconditioner with a tolerance of $1e-7$. The solution in this case is also not guaranteed to be exact.
- **LU** (Lu decomposition): this method decomposes the information matrix into two triangular matrices, one of them with values in the upper right diagonal, and the other one on the lower left diagonal. It is an exact solution and works by solving the system for each of the triangular matrices.
- **Cholmod2** (supernodal sparse Cholesky backslash): this is not a method per se but an implementation of a solution using the Cholesky decomposition for sparse matrices. It is similar to the LU decomposition but in this case the two triangular matrices are the tranpose of each other. It is part of the package SuiteSparse [2] by Tim Davis which is available online.

The essential advantage of the information form is that the information matrix contains many elements that are close to zero. In order to gain computational speed, it is possible to approximate these numbers by zero, in other words, the information matrix can be sparsified. Depending on the amount of "close to zero" elements in the information matrix and their distance to the zero approximation, this sparsification can lead to big errors, hence making the sparsification not useful.

[2]SuiteSparse is a collection of packages for working with extremely large sparse matrices. It is freely available online at: http://www.cise.ufl.edu/research/sparse/SuiteSparse/

2.4 Filter Consistency

The estimation of the state of the process in the EKF and EIF is characterized by its uncertainty which is also estimated. One of the issues that can arise when estimating both the process and its uncertainty is the fact that the uncertainty might not be sufficiently close to the real uncertainty. The EKF models the uncertainty of the state estimate based on the uncertainty of both the dynamics and the measurements. If the estimated uncertainty is deviated from the true error with respect to the ground truth data, there exists an inconsistency between the uncertainty of the measurements and the dynamics and the uncertainty of the estimated state.

The fact that the state of the process might not be accurate is not as relevant as long as the estimation of the uncertainty is accurate (i.e. we know how much error we have). The problem arises when the filter becomes optimistic (under estimating the uncertainty) or pessimistic (over estimating the uncertainty).

The reason for which the filter becomes either optimistic or pessimistic is the error that arises from the linearization of the measurement equation during the linearization procedure (remember that in our set up, the control equation remains linear hence no linearization is performed). As the true distribution cannot be fully described with a Gaussian model (though how big the error is depends very much on the type of measurements we take), the error on the linearization accumulates at every time step up to the point where the filter can no longer estimate accurately the error on the estimation of the state.

There are several test that can be performed to analyze when and why a filter becomes inconsistent. For these tests to succeed a ground truth state is usually required, however, they can be very insightful in simulated data in order to understand the fundamental information that the measurements convey in the measurement update step.

The *consistency criteria* of a filter are as follows [1]:

1. The state error should be acceptable as zero mean and have magnitude commensurate with the error covariance estimated by the filter (NEES)
2. The innovations should also have the same property (NIS)
3. The innovations should be acceptable as white (Whiteness)

Only the last two criteria can be tested on data where a ground truth state and error is not known (typically real data). The first criterion, which is the most important one, can only be tested when the ground truth state is available. In our work we do not test to Whiteness.

NEES

The **normalized (estate) estimation error squared** (NEES) test is meant to test if the filter satisfies the **criteron 1** of the filter consistency criteria.

The NEES is defined as:

$$\epsilon_k = (\check{x}_k - x_k)^T \times P_k^{-1} \times (\check{x}_k - x_k), \tag{2.112}$$

where $\check{x}_k$ is the ground truth state.

The test is based on the results of *Monte Carlo Runs* that provide N independent samples of the random variable ϵ_k. The sample average NEES for N montecarlo runs is then:

$$\bar{\epsilon}_k = \frac{1}{N}\sum_{i=1}^{N}\epsilon_k^i. \tag{2.113}$$

Then $N\bar{\epsilon}_k$ will have, under the hypothesis H_0 that the filter is consistent, a chi squared density with Nn_x degrees of freedom (n_x is the dimension of the state x). Under these circumstances the first criterion will be accepted if:

$$\bar{\epsilon} \in [r1, r2], \tag{2.114}$$

where the acceptance interval $[r1, r2]$ is determined such that

$$P\{\bar{\epsilon}_k \in [r1, r2] \mid H_0\} = 1 - \alpha, \tag{2.115}$$

where $1 - \alpha$ represents the confidence value (typically 95%).

This test can also be used for a single run, though the more runs are used the narrower the acceptance interval becomes.

A bias in the state estimation error will increase ϵ_k which will yield unacceptable values for the statistic $\bar{\epsilon}_k$ (i.e. it will be outside the acceptance interval).

NMEE

If the NEES test is not satisfied, a separate bias test can be carried out to identify the source of the problem. This is done by taking each component of the state error $\check{x}_k - x_k$ divided by its standard deviation (its estimated error covariance) and checking whether its mean be accepted as zero.

The statistical test for the normalized mean estimation error (NMEE) for component j of the state from runs $i = 1, ..., N$ is:

$$(\bar{\mu}_k)_j = \frac{1}{N}\sum_{i=1}^{N}\frac{(\check{x}_k - x_k)_j^i}{(P_k)_{jj}}. \tag{2.116}$$

The acceptance interval in this case, having an acceptance region of 95%, is defined as:

$$(\bar{\mu}_k)_j \in [-r, r], \tag{2.117}$$

where $r = \frac{1.96}{\sqrt{N}}$

NIS

The same procedure as with the NEES can be carried out for the innovations $\nu_k = z_k - \hat{z}_k$ (**criterion 2**). Under the hypothesis that the filter is consistent, the normalized innovation squared (NIS) is defined as:

$$(\epsilon_k)_\nu = \nu_k^T \times (J_H \hat{P}_k J_H^T + R)^{-1} \times \nu_k. \tag{2.118}$$

Note that this test can be carried out with no ground truth available, however it is performed under the hypothesis that the filter is consistent.

Using N independent samples, the average NIS is calculated as:

$$(\bar{\epsilon}_k)_\nu = \frac{1}{N} \sum_{i=1}^{N} (\epsilon_k)_\nu^i, \tag{2.119}$$

which is tested in the same way as the NEES test but with an acceptance region based on the fact that $N(\hat{\epsilon}_k)_\nu$ is chi-squared distributed with Nn_z degrees of freedom, where n_z is the dimension of the measurement vector z.

Chapter 3

Omnivision Trajectory Based SLAM

So far we have discussed the general ideas behind the KF, its extended version, the EKF, for non-linear processes, and the alternative representation in the EIF. We are now particularly interested in the application of such methods to the estimation of a robotic process where a robot moves in a certain environment and using its internal sensors it is capable of estimating its relative pose while it builds a map of the environment (SLAM problem). If we consider both the robot pose and the map of the environment as a process governed by a control equation (the robot moving) and a measurement equation (the robot sensing) it is easy to understand how such a representation is parallel to the state estimation we discussed before.

Within the classical Kalman Filter SLAM framework and its derivations the EKF and EIF, multiple approaches have been presented with respect to both the features that are extracted (used as measurements from the environment) and the representation of the state vector that is going to be estimated. The standard EKF SLAM solution estimates landmark positions, the map, and the current robot pose. In most visual SLAM algorithms, a limited set of landmarks is used in order to reduce the computational costs (the more features, the bigger the state vector and covariance estimation, hence the more complex the solution).

We present a combination of two methods for both feature extraction and state representation that present a clear advantage with respect to more classical approaches. Eustice et al. [7] present an elegant solution using the EIF where the state is represented as the current robot pose and the set of all previous poses. This approach for the state representation is called *Trajectory Based SLAM*. Based on this representation, the measurement of the environment is directly affected as we now need to extract features that are related to the state representation. Using an Omnivision image taken at every robot pose we take advantage of the extraction of image features and epipolar geometry to obtain information regarding the relative poses between pairs of images.

3.1 Trajectory Based State Vector

In conventional visual landmark based SLAM the state $\mathbf{x}$ of the estimation process consists of only the most recent robot pose as well as a growing set of 2D or 3D positions of landmarks reconstructed from the camera, which is called the map. During every update newly seen landmarks are added to the state vector (new features of the map) and the robot pose is updated to the last pose. In Trajectory Based SLAM the landmarks are not explicitly modeled, rather, **the state** at time step k, x_k, consists of current and all previous robot poses, which in our case are the 2D positions and orientation angles:

$$x_k = \begin{bmatrix} \dot{\mathbf{x}}_k^* \\ M \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{x}}_k^* \\ \dot{\mathbf{x}}_{k-1} \\ \vdots \\ \dot{\mathbf{x}}_0 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_k \\ \mathbf{y}_k \\ \theta_k \\ \mathbf{x}_{k-1} \\ \mathbf{y}_{k-1} \\ \theta_{k-1} \\ \vdots \\ \mathbf{x}_0 \\ \mathbf{y}_0 \\ \theta_0 \end{bmatrix}, \tag{3.1}$$

where $\dot{\mathbf{x}}_k$ is the 3D vector containing the robot pose at time step k ($\mathbf{x}$, $\mathbf{y}$ and heading θ) and the first pose in x_k is always the current robot pose, also called $\dot{\mathbf{x}}_k^*$ while the remaining of the poses represent the map M. This representation has a direct implication in the complexity of the EIF as we will discuss in the next section.

Using the raw sensor data and the estimated robot trajectory (the state), a representation of the environment can be built at every moment during the estimation process.

Figure 3.1 visualizes an example operation of the Trajectory SLAM with 4 robot poses.

- The first robot pose $\dot{\mathbf{x}}_0 = [\mathbf{x}_0, \mathbf{y}_0, \theta_0]^T$ is added to the state as $[0, 0, 0]^T$ and thus defines the coordinate frame for the rest of the robot poses.
- An omnidirectional image is taken at this position and stored in memory.
- **Time Update Step:** The next robot pose $\dot{\mathbf{x}}_1$ is added to the state using the odometry readings $\mathbf{u}_0$ (the control vector).
- **Measurement Update Step:** An image is taken at the current robot pose which is then compared with the image from the previous pose providing additional positional information $\mathbf{z}_1^0$, the observation vector, that is used to improve the estimate of the state.

This procedure is repeated for every new robot pose. In $\dot{\mathbf{x}}_2$ a new image is taken and matched with both previous robot poses. Then the robot drives around the corner to $\dot{\mathbf{x}}_3$, causing the overlap of the new image with the first two images to reduce. The image comparison method, as explained in in the following section, detects this and the pose estimation of $\dot{\mathbf{x}}_3$ will be based on the observation of the last pose $\mathbf{z}_3^2$ and the odometry reading $\mathbf{u}_3$.

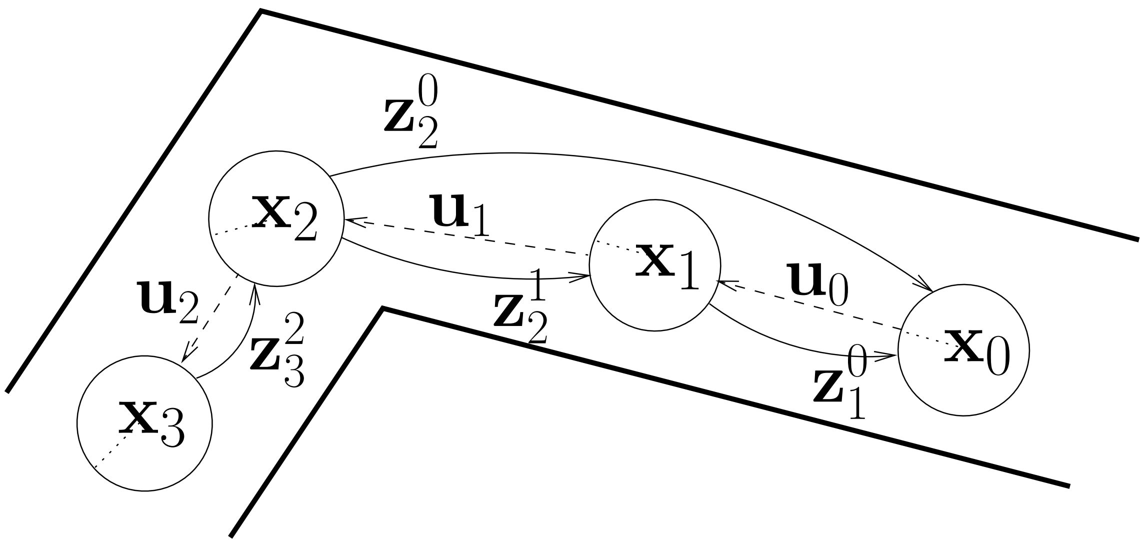

Figure 3.1: Example of state augmentation procedure

This example is sufficiently clear to illustrate the procedure that the trajectory based SLAM follows to create the map using previous robot poses as environment landmarks. There is however an important aspect of SLAM that is not addressed in this example, the loop closing problem.

3.2 Omnivision Based Measurement Vector

The measurement vector z_k contains the information extracted from the environment at time k that is used in the measurement update step of the KF to improve the estimation of the state x_k. We approach the problem of measuring the environment by taking omnidirectional images at every robot pose and measure the relative pose between the last image and all the previous ones whenever it is possible. Thus, an observation $\mathbf{z}$ describes the relative positional information extracted from the current omnidirectional image and all the previous images which depict the same part of the environment.

It is well known that the relative pose can be estimated from two images using the epipolar constraint [10]. This estimation is performed by first extracting a set of salient local image features from both omnidirectional images which are then compared to give a set of n 3D point correspondences, $\mathbf{p}_1, \ldots, \mathbf{p}_n$ on one image surface and $\mathbf{q}_1, \ldots, \mathbf{q}_n$ on the other. In our experiments we used SIFT features (Scale Invariant Feature Transform) [15]. Point correspondences that resulted from the same world point in the environment can be related by the essential matrix which describes the relative camera pose:

$$\mathbf{p}_i^T E \mathbf{q}_i = 0 \text{ , for all } i. \tag{3.2}$$

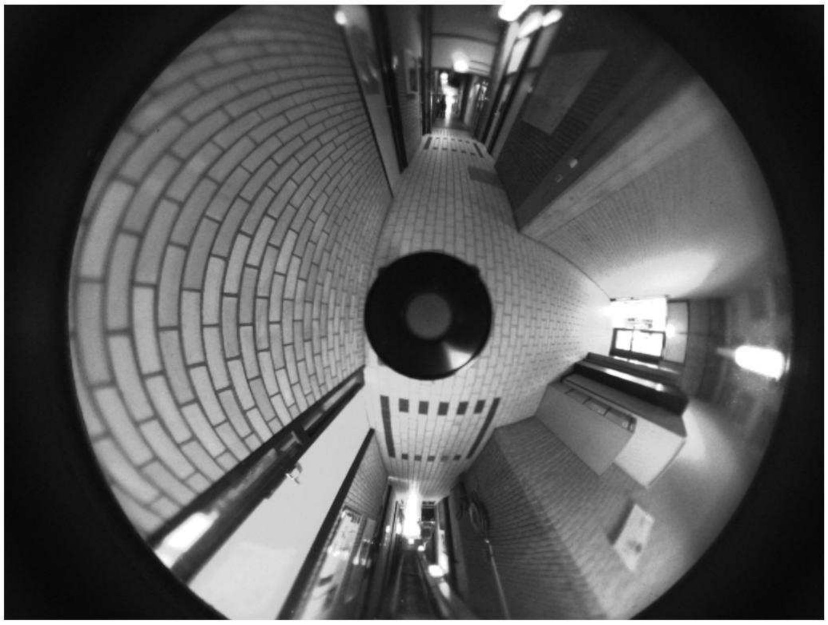

Figure 3.2: Sample Image taken with the Omnivision Camera

Based on this function, the matrix E can be estimated from 8 correspondences using the 8-point algorithm [10]. If we take into account that in our environment setup the robot moves on a planar surface the estimation of E can be done using only 3 correspondences by applying the similar 3-point algorithm[4].

To be robust against false point correspondences we use the 3-point algorithm inside the hypothesize and test method RANSAC (RANdom SAmple Consensus). This provides us with the matrix E and the number of correspondences for which the reprojection error given E is small. If the ratio between this number and the number of features found in the images is bigger than a certain threshold, which we set to 0.1 in our experiments, then we extract the pose information from E. Otherwise, we do not use this image pair for the observation. In this way we use the same algorithm for both determining which measurements to add to the observation vector, solving the data association problem, and computing the measurements themselves.

From E the relative pose can be extracted using [11] and results, in the case of 2D motion, in the heading of the translation ϕ and a 2D rotation θ. As opposed to landmark based SLAM, where only the direction of the observed landmarks is taken into account to improve the state estimate, for Trajectory based SLAM, both these two parameters are used. The observation vector at time step k thus gets the following form:

$$z = \begin{bmatrix} \phi_s \\ \theta_s \\ \phi_{s-1} \\ \theta_{s-1} \\ \vdots \end{bmatrix}, \tag{3.3}$$

where the set of s observation pairs $[\phi, \theta]$ correspond to the set of omnidirectional images, taken at previous robot poses, for which a sufficient set of feature matches were found.

3.3 Exactly Sparse Extended Information Filter

As we discussed before, the EIF represents some advantages with respect to the EKF due to the possibility to perform certain sparsification over the information matrix. Such sparsification, depending on the state representation and the measurement method used, will yield a certain amount of error that will be transfered from step to step.

As presented by Eustice. et al. [7], a trajectory based approach in the EIF SLAM method presents some natural advantages with respect to other classical state representations where a set of natural landmarks are present in the state vector. Namely, for such a representation, the information matrix becomes *naturally* sparse, hence no artificial sparsification is required and no error is induced on the estimation results. As we shall see, the natually sparse characteristic of the information matrix for a trajectory based SLAM approach allows the mapping and localization problem to be addressed in *almost* linear time.

In this section, we first present a brief description of the state augmentation procedure in the EKF (how the state vector increases in size as the robot moves and new poses are introduced as part of the trajectory). Then, using the same augmentation scheme, we present the state augmentation procedure for the information representation of the solution, the EIF. We also analyze the resulting information matrix and explain its naturally sparse property. Finally, we draw some conclusions regarding the gain in computational costs for the trajectory based SLAM approach.

3.3.1 State Augmentation in the EKF

As the EKF assumes a standard parametrization of the Gaussian distribution, and the process is linearized by means of a first order approximation, we know that the estimate at time step k is described by a Gaussian distribution [1] as follows:

$$p(\dot{\mathbf{x}}_k^*, M \mid z_k, u_k) \quad = \quad \mathcal{N}\left(\begin{bmatrix} \mu_{\dot{\mathbf{x}}_k^*} \\ \mu_M \end{bmatrix}, \begin{bmatrix} \Sigma_{\dot{\mathbf{x}}_k^* \dot{\mathbf{x}}_k^*} & \Sigma_{\dot{\mathbf{x}}_k^* M} \\ \Sigma_{M \dot{\mathbf{x}}_k^*} & \Sigma_{MM} \end{bmatrix} \right)$$

[1] The mean μ and covariance Σ of the distribution represent the state x and error covariance P estimated in the EKF.

This distribution represents the map M and robot pose $\dot{\mathbf{x}}_k^*$ given the measurements z_k and control input u_k. When a new robot pose is reached (before new measurements are obtained, hence time update step), the robot state needs to be augmented to include the new pose in the map and get a new estimate for the robot state, leading to the new distribution:

$$p(\dot{\mathbf{x}}_{k+1}^*, \dot{\mathbf{x}}_k, M \mid z_k, u_{k+1}) \quad = \quad \mathcal{N}\left(\mu'_{k+1}, \Sigma'_{k+1}\right) \tag{3.4}$$

where the augmented mean and covariance (a priori state and error covariance) are defined based on the first order approximation as:

$$\begin{aligned}
\mu'_{k+1} &= \begin{bmatrix} \mu_{\dot{\mathbf{x}}_k^*} + u_{k+1} \\ \mu_{\dot{\mathbf{x}}_k^*} \\ \mu_M \end{bmatrix} \\
\Sigma'_{k+1} &= \begin{bmatrix} (\Sigma_{\dot{\mathbf{x}}_k\dot{\mathbf{x}}_k} + Q) & \Sigma_{\dot{\mathbf{x}}_k\dot{\mathbf{x}}_k} & \Sigma_{\dot{\mathbf{x}}_k M} \\ \Sigma_{\dot{\mathbf{x}}_k\dot{\mathbf{x}}_k} & \Sigma_{\dot{\mathbf{x}}_k\dot{\mathbf{x}}_k} & \Sigma_{\dot{\mathbf{x}}_k M} \\ \Sigma_{M\dot{\mathbf{x}}_t} & \Sigma_{M\dot{\mathbf{x}}_k} & \Sigma_{MM} \end{bmatrix},
\end{aligned} \tag{3.5}$$

which is an equivalent and extended representation of the a priori error covariance estimation found in the algorithm of the EKF (see algorithm 2.2.3).

3.3.2 State Augmentation in the EIF

The EIF is based on the alternative representation of a Gaussian distribution, the canonical or information form. Using this representation, we know that the estimate at time k is described by an information form of the Gaussian distribution as:

$$p(\dot{\mathbf{x}}_k^*, M \mid z_k, u_k) \quad = \quad \mathcal{N}^{-1}\left(\begin{bmatrix} \eta_{\dot{\mathbf{x}}_k^*} \\ \eta_M \end{bmatrix}, \begin{bmatrix} \Lambda_{\dot{\mathbf{x}}_k^*\dot{\mathbf{x}}_k^*} & \Lambda_{\dot{\mathbf{x}}_k^* M} \\ \Lambda_{M\dot{\mathbf{x}}_k^*} & \Lambda_{MM} \end{bmatrix}\right)$$

Again, when a new robot pose is reached the robot state needs to be augmented:

$$p(\dot{\mathbf{x}}_{k+1}^*, \dot{\mathbf{x}}_k, M \mid z_k, u_{k+1}) = \mathcal{N}^{-1}\left(\eta'_{k+1}, \Lambda'_{k+1}\right) \tag{3.6}$$

Taking the previous standard representation of this distribution used in the EKF, and the formal relation between the normal form and the information form (see Appendix A.1), it is possible to reach a state augmentation scheme by means of inverting the augmented error covariance Σ'_{k+1} (equation 3.5), obtaining the information vector η'_{k+1} and information matrix Λ'_{k+1} (see Section 2.3.2):

$$\begin{aligned}
\eta'_{k+1} &= \begin{bmatrix} Q^{-1}(u_{k+1}) \\ \eta_{x_k} - Q^{-1}(u_{k+1}) \\ \eta_M \end{bmatrix} \\
\Lambda'_{k+1} &= \begin{bmatrix} Q^{-1} & -Q^{-1} & 0 \\ -Q^{-1} & \Lambda_{x_k x_k} + Q^{-1} & \Lambda_{x_k M} \\ 0 & \Lambda_{M x_k} & \Lambda_{MM} \end{bmatrix}
\end{aligned}$$

Note that the matrix Q is rather simple and somehow different than the one defined for the EKF. In this case, the matrix Q does only contain the error model for the odometry, hence it is a simple 3×3 matrix.

3.3.3 Sparse Information Matrix

Note the zeros that result from augmenting the state in the information form. This is the key result that leads to the computational gain (naturally sparse) of the EIF when using a trajectory-based approach. When the state vector is augmented to include the new robot position $\mathbf{x}_{t+1}$ only shared information between the robot state and the previous robot state is introduced and the shared information between the map M and the new robot pose is always zero. If we would continue to introduce new states, we shall observe that the information matrix will present a block tridiagonal structure where each state is only linked to the previous and following one. The only exception occurs when a loop is closed. In such situation, non diagonal elements will appear in the information matrix as shared information between the new robot state and previously visited robot states is introduced.

When we marginalize the state $\mathbf{x}_t^*$ from the distribution in equation 3.6 to perform the time update step, it can be proved [7] that it can be implemented in constant time as only a fixed portion of the information matrix is involved in the marginalization calculation.

Having seen the state augmentation in the time update step in both the covariance and information form, we can similarly obtain the expression in the information form for the measurement update step [7]:

$$\begin{aligned}
\eta_t &= \hat{\eta}_t + {J_H}^T R^{-1}(\mathbf{z}_t - h(\hat{\mu}_t) + H\hat{\mu}_t) \\
\Lambda_t &= \hat{\Lambda}_t + {J_H}^T R^{-1} J_H
\end{aligned}$$

This description of the measurement update step in the information form shows that the information matrix is additively updated by the product $H^T R^{-1} H$. As the jacobian H is always sparse [7] only some elements of the information matrix need to be modified, hence the updates are *constant in time.*

Up to this point, the total complexity of the information filter (the algorithm alone regardless of the state recovery) is constant in time (both prediction and measurement) as opposed to the cubic complexity of the standard EKF. The state recovery process can be carried out in almost linear time with the help of Cholesky decomposition as implemented in the CHOLMOD package we used in out Matlab experiments, so the global computation of the filter grows almost linearly with the size of the map.

Chapter 4

Experimental Results

4.1 Introduction

Our approach to the SLAM problem did face three different challenges: computational complexity of the standard EKF solution, adequacy of an Omnivideo camera to measure the environment and the loop closing problem. In order to test the performance of our Omnivision Trajectory Based EIF approach, we carried out a set of experiments to demonstrate how our approach can solve each of the three problems.

Three set of experiments are discussed. The first experiment aims at the illustration of the inconsistencies that arise during the estimation process due to the errors in the linearization of the measurement process. The second experiment illustrates the important computation gain of the EIF with respect to the standard EKF solution and also depicts the ability of our approch to build consistent maps of small environments. The third experiment is to our knowledge the largest experiment carried out in Visual SLAM and it was designed to measure the ability of our method to close large loops where the odometry error is very large.

4.2 Experimental Setup

For our experiments we used a Nomad Scout (see figure 4.1) robot equipped with an Omnivideo system consisting of a one megapixel firewire camera and an Accowle convex hyperbolic mirror [24]. Additionally, and for visualization purposes only, the robot was equipped with a laser range scanner. The measurements taken with the laser were then used after the trajectory was corrected with our method to visualize the environment and illustrate the improvement in the accuracy of the map.

Due to the large field of view of the camera it was possible to generate 360 degrees images (see figure **??**). Two images per second were recorded while the robot was driving at a maximum speed of 20 cm per second, resulting in an average of 1 image every 10 cm. A large portion of the long

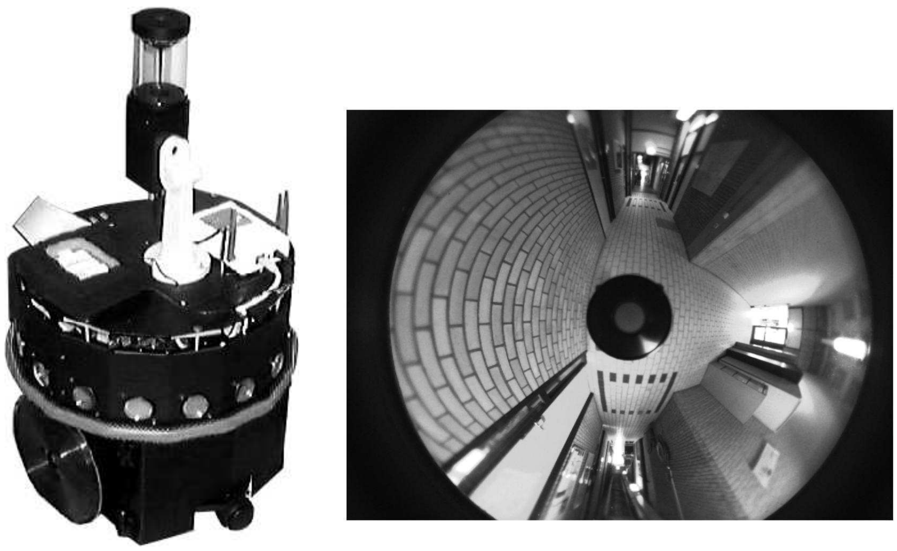

Figure 4.1: LEFT: Nomad Scout Robot with Omnivision System - RIGHT: 360 degrees image

hallways where the robot was driven were poorly iluminated, posing a real challenge for the image matching algorithm. We measured the amount of light in some of these corridors obtaining an equivalent amount of light to a living room lit by Christmas tree lights (30 lux).

4.3 Artificial Data Set

As an illustration of the inconsistency that inevitably arises in any non linear process when approximated with a linear function, we created an artificial dataset. It consisted of a trajectory where the robot drives 4 times around a circle in the counterclockwise direction (see figure 4.2). Each lap around the circle consisted on 50 robot poses where the robot was able to observe all previous poses that lay within a certain radius of the current pose. As simulated data does not contain the Omnivision images, we also simulated the measurements (relative poses) that real images will yield, making our dataset completely compatible with our method.

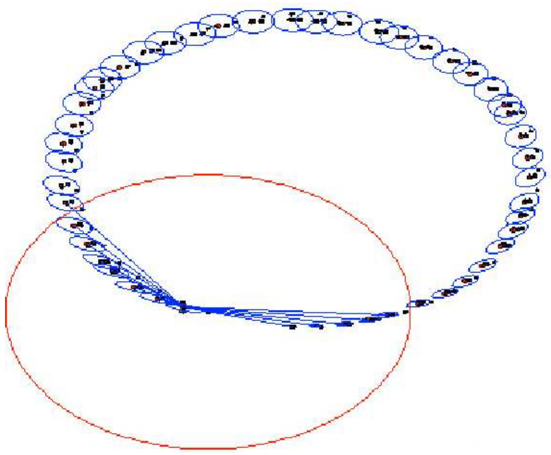

Figure 4.2: Artificial data set where the robot drives following a circular trajectory. The small circles around each robot pose depict the error covariance of each pose and are centered on the estimated robot pose. The large circle represents the observable area. Ground truth is represented by the remaining points with no circle around.

The purpose of this experiment is the illustration of the accumulation of error in the estimation of both the state and the error covariance due to the linearization, hence the appearance of inconsistency at certain point during the estimation. The reason to use an artificial data set is the fact that an artifical data set also contains the ground truth, hence the filter consistency can be measured (see section 2.4)

The results of this experiment can be appreciated in figure 4.3. The first plot represents the NEES measurement, meant to test if the state error should be acceptable as zero mean and have magnitude commensurate with the error covariance estimated by the filter. We can see how the filter becomes too optimistic after a few steps. An optimistic deviation occurs when the estimated error covariance is smaller than it should be. This is also observable in the trajectory plot (see figure 4.2 where we can see how the estimated poses get out of the circles representing the error covariance as the robot moves along the trajectory). The most interesting observation in this plot is the fact that the NEES measure drops at exactly the loop closing points. This behavior (the filter becoming consistent again) ocurs because the robot was observing position zero, which is by definition the most accurate one as it is recorded as a position with zero error.

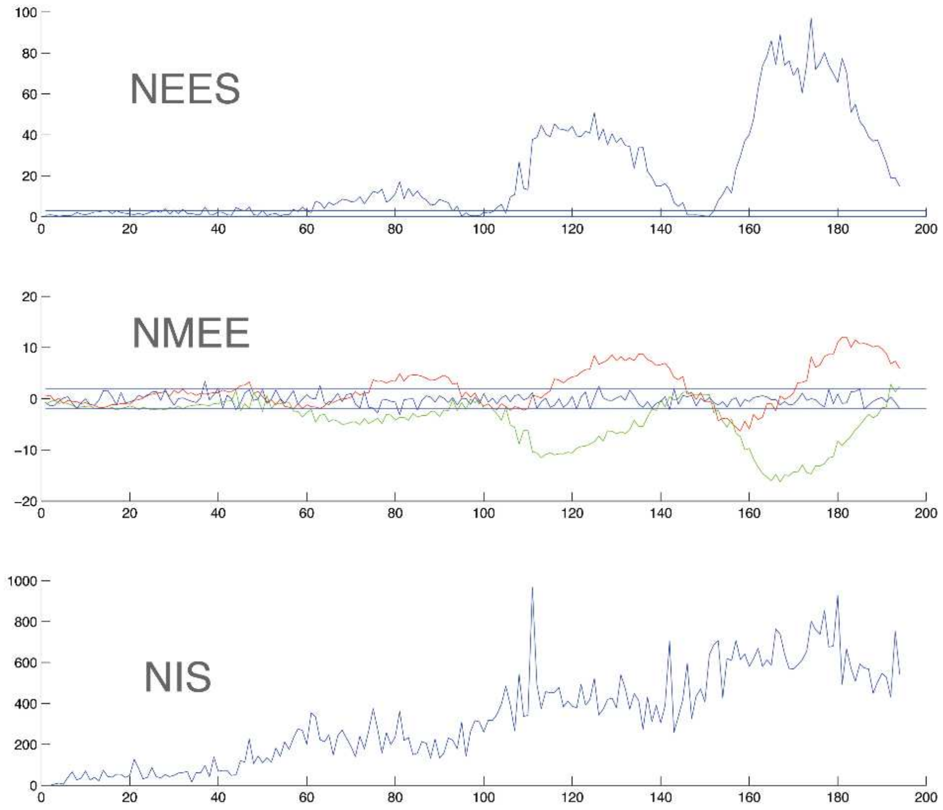

Figure 4.3: NEES, NMEE and NIS for an artificial data set

The second plot, NMEE, is meant to test the same properties on the innovations. Each of the three lines represents one dimension in the robot pose, X position, Y position and heading α Again, we see how the filter becomes inconsistent after a few iterations, and how this inconsistency diminishes at the loop closing points. One particularly interesting observation is that the NMEE test is fully consistent for the heading dimension. We believe that this behavior is induced by the circular shape of the trajectory as it was only present in our simulations with perfectly circular trajectories.

Lastly, the NIS test depicts how the state grows over time, showing significant increases at the loop closing points. This is induced by the fact that at the loop closing points, the amount of observed robot poses increases by a factor equal to the lap around the circle.

4.4 Small Office Environment

One of the most challenging environments to perform visual SLAM is an office space. The major difficulty that any visual SLAM method faces in office environment is the visual similarities that many hallways and offices share. These similarities make image matching difficult (see figure 4.4). By using omnivision images along with a robust image matching algorithm to perform SLAM over a map we prove that our approach is more than adequate for office environments even when odometry (very accurate usually) yields enormous errors in the long run.

This experiment was preformed in an office environment where the robot was driven manually around a small loop where two differente offices were visited along with a small portion of the hallway that connects them. The trajectory was driven twice in order to test our approach when previously seen areas are revisited. The total trajectory consist of 875 images taken at every robot pose along with odometry measures (displacement in x, displacement in y and heading of the robot α). Due to the smooth surface and the limited size of the environment, we artificially increased the odometry error at some points (by artificially making the robots wheel slip) during the trajectory to better illustrate the improvement in the accuracy of the SLAM corrected map.

We present two maps of the same trajectory. Each map contains the same information, namely, the estimated trajectory of the robot (black circles), the laser data obtained at every robot location and the connected graph that represents the images that were found to have sufficient similarity (light gray lines connecting robot poses).

As we can see in figure 4.7, the sections of the trajectory where the robot drives multiple times over a hallway or office where no occlusion is present, the images taken at those poses are correctly matched (plenty of links between those poses in the map). In fact, we can see that no false links (links crossing walls for instance, or links across far away poses) are presents in this dataset.

The first map (see figure 4.7) was built using odometry as the only information source. Despite the fact that some structure can be distinguished in the map (walls, doors and hallways), the odometry error rapidly adds up yielding duplicated structures (see figure 4.5, Left). If the robot would continue driving the same space for more loops, the accumulated error will make the map completely cluttered.

The second map (see figure 4.8) represents the same trajectory corrected with our Omnivision Trajectory-based SLAM approach. Two essential aspects are shown in this map. Firstly, it is clear

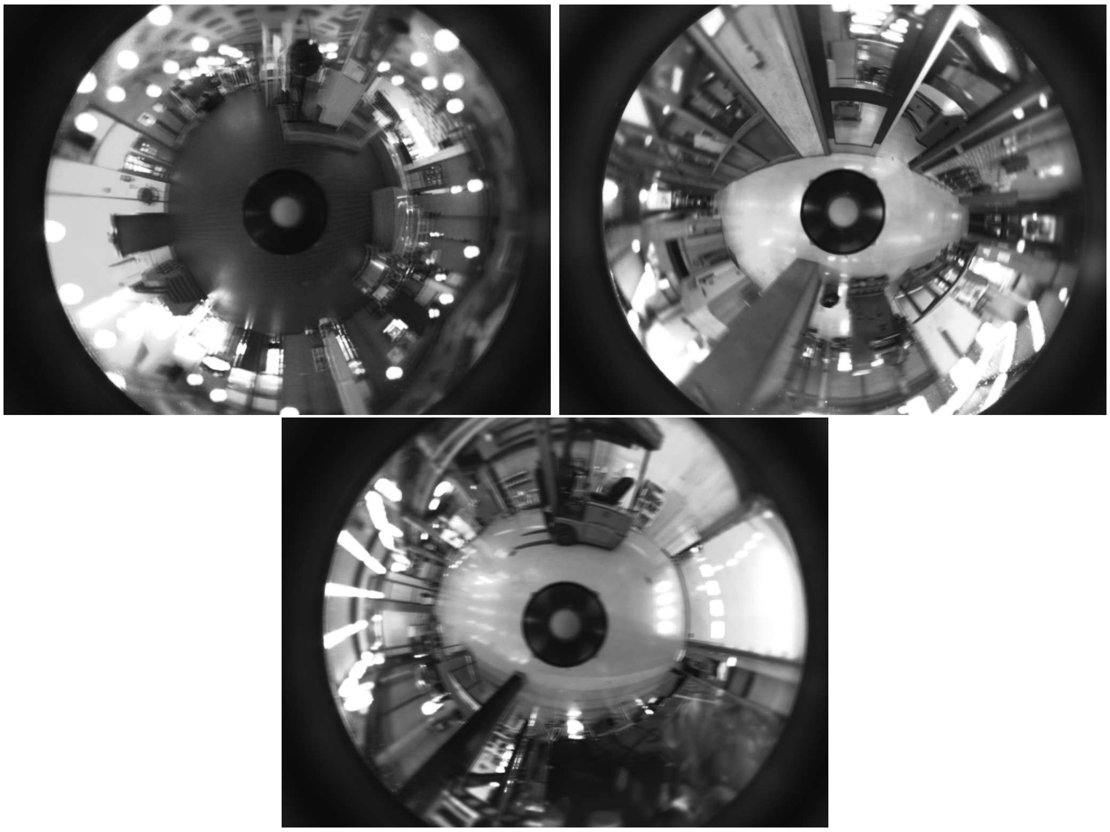
Figure 4.4: Typical hallway images

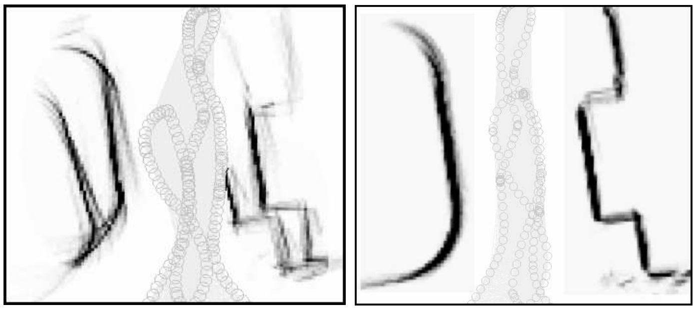
Figure 4.5: **Left**: Odometry map, **Right**: SLAM map - small office

that our method can perfectly cope with loop closure in small environments as the duplication of structures (laser data) in loop closing points is no longer present (see figure 4.5, Right) and previously seen areas are correctly matched together. Secondly, the fact that no duplicated structures are present at all in the corrected map implies that the information regarding the relative pose obtained through the omniview images convey enough information to compensate for the accumulation of error in odometry. Features like walls, hallways, doors and even the small "box-like" structure where the robot goes around are now clearly visible.

Regarding the representation of the information matrix, the diagonal structure discussed in section 3.3 can be appreciated (see figure 4.6). The non diagonal elements represent the information introduced when a loop is closed. The small non diagonal elements crossing perpendicularly the main diagonal represent the information introduced when small loops are closed (for instance when the robot walks around the box-like structure). The other non diagonal elements (far away from the main diagonal but in the same direction) represent the information introduced when the big loop is closed (the robot comes back to the initial position. It is important to note the clear difference between the information matrix and the error covariance matrix. In the information matrix, new information is only introduced as links between the previous pose and the following one and only additional information is present in the case of loop closure. On the other hand (see figure 4.6), the correlations present in the error covariance matrix are updated at every stage and the matrix presents a "checkers board-like" structure. The fact that the information matrix presents so many "white" space (actual zeros in the matrix) is a fundamental gain in computational complexity as only a few elements in the matrix are updated at every step, hence the matrix is naturally sparse.

[1]

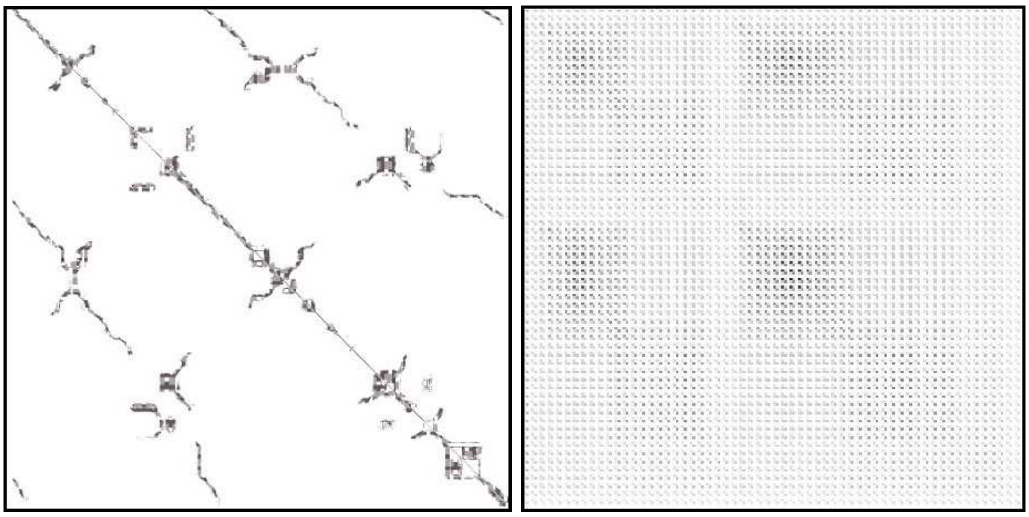

Figure 4.6: **Left**: Information matrix, **Right**: error covariance matrix - small office

[1] In our experiments we also run the standard EKF solution to compare obtaining indeed the exact same results.

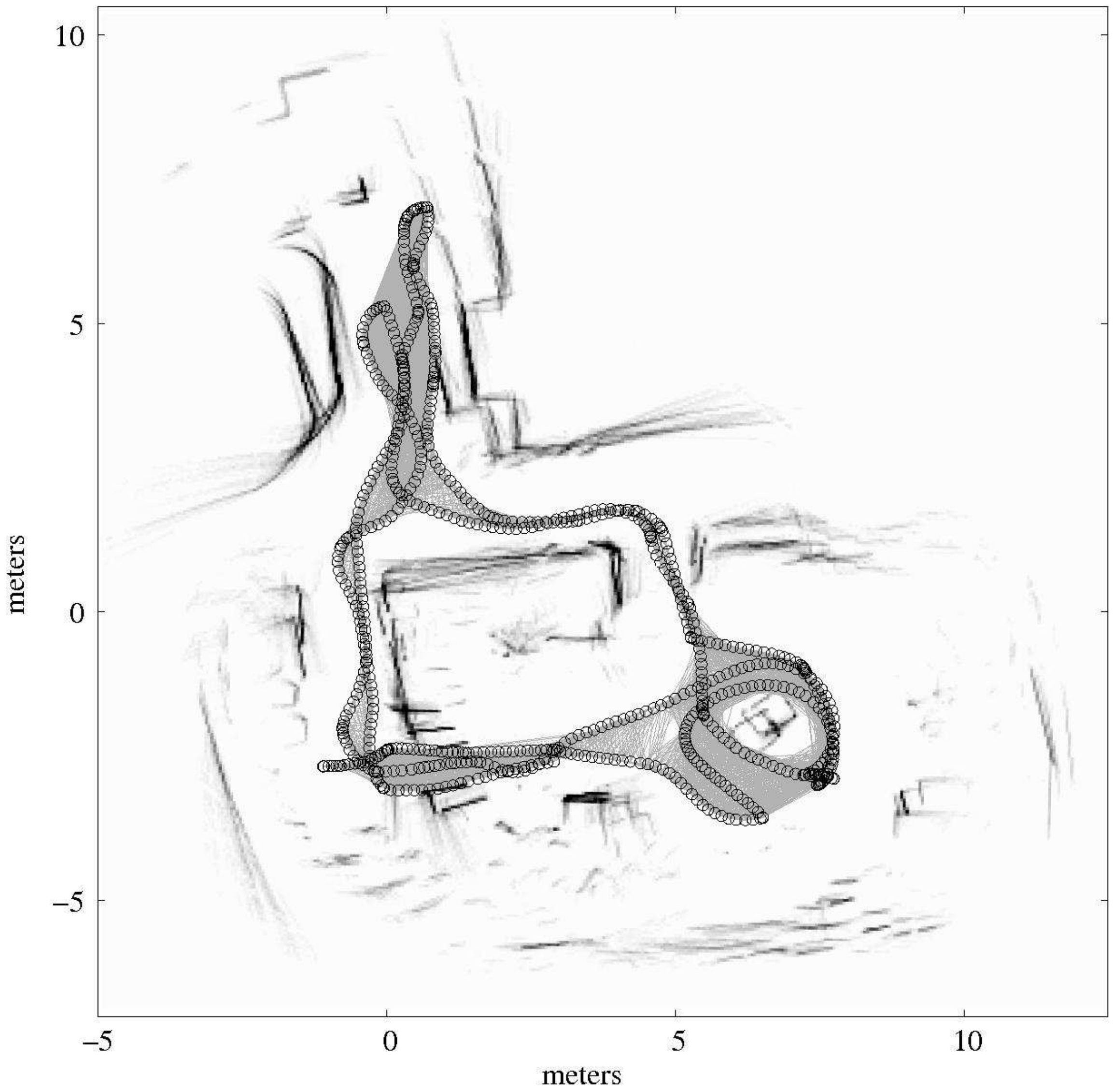

Figure 4.7: Odometry based map with laser data and connectivity map for robot poses - small office

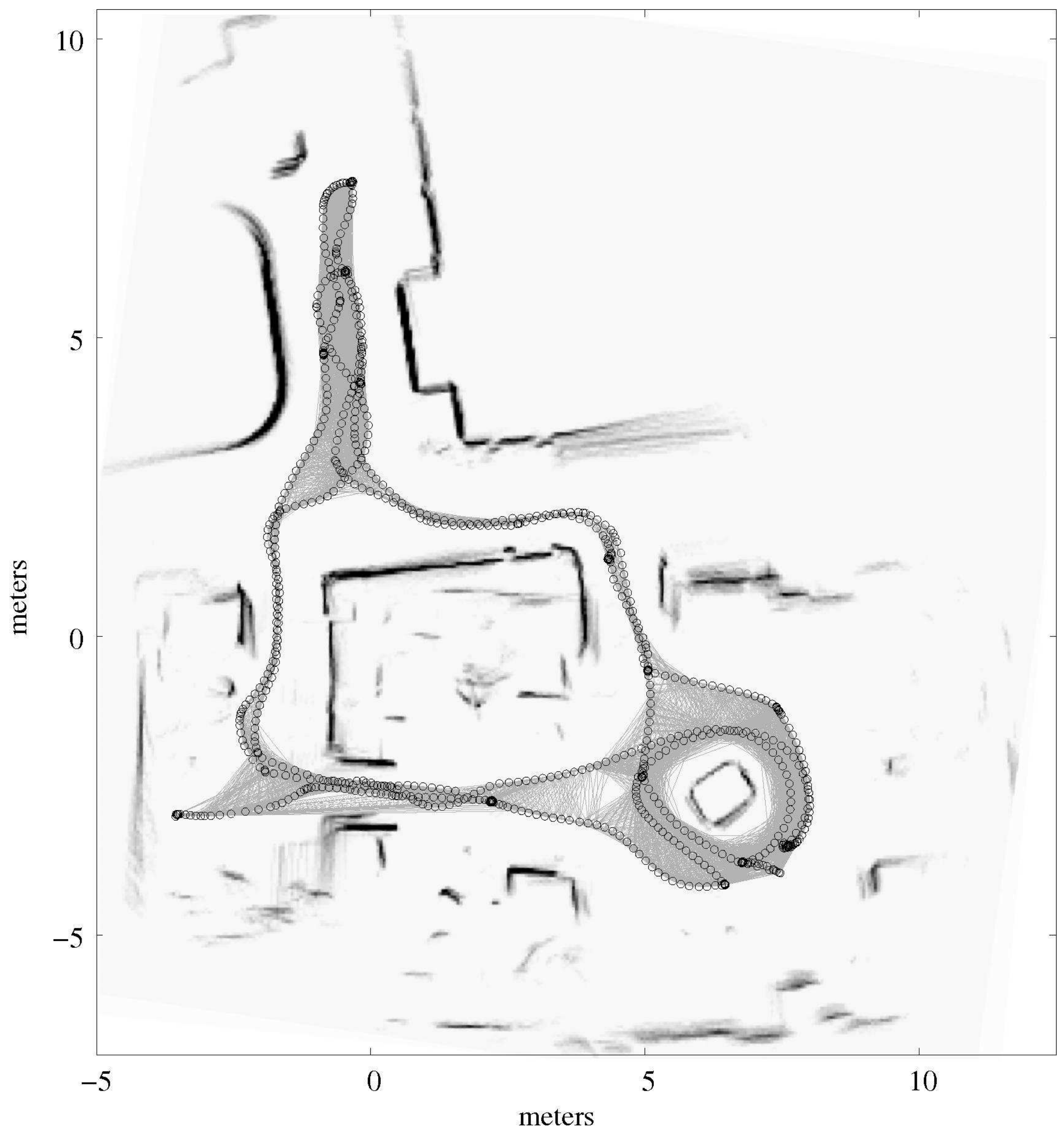

Figure 4.8: Omnivision SLAM corrected map with laser data for visualization purposes - small office

4.4.1 Complexity

To finalize our set of experiments, and to test the computational gain of the EIF versus the golden standard EKF, we analyzed the computing time (see figure 4.9) of the SLAM algorithm over the small office dataset. The complexity of the ESEIF is at most quadratic including the state recovery process (quadratic at most, but can be approximately linear using Cholesky decomposition), however, the computational cost would rise linearly in the number of observations (in practice this can be considered constant) if the state recovery is not included [7]. This is enough to extend the viable use of the filter well beyond the 1000 steps, showing a reduction ratio in the construction of the complete map of 6 times. During the experiments, we run the SLAM algorithm offline using both the EKF and ESEIF and accounted for the computation time of both. One of the interesting aspects of the figure below (see figure 4.9) is the peaks that can be observed in the EKF computation speed. As we discussed earlier, the more image matches are found for one time step, the more information is updated in the error covariance matrix, hence the more computation time is needed. The high times present in the EKF line represent moments in the trajectory where plenty of observations where present (namely, the small loop around the "box-like" structure and the portion of the hallway shown in figure 4.5).

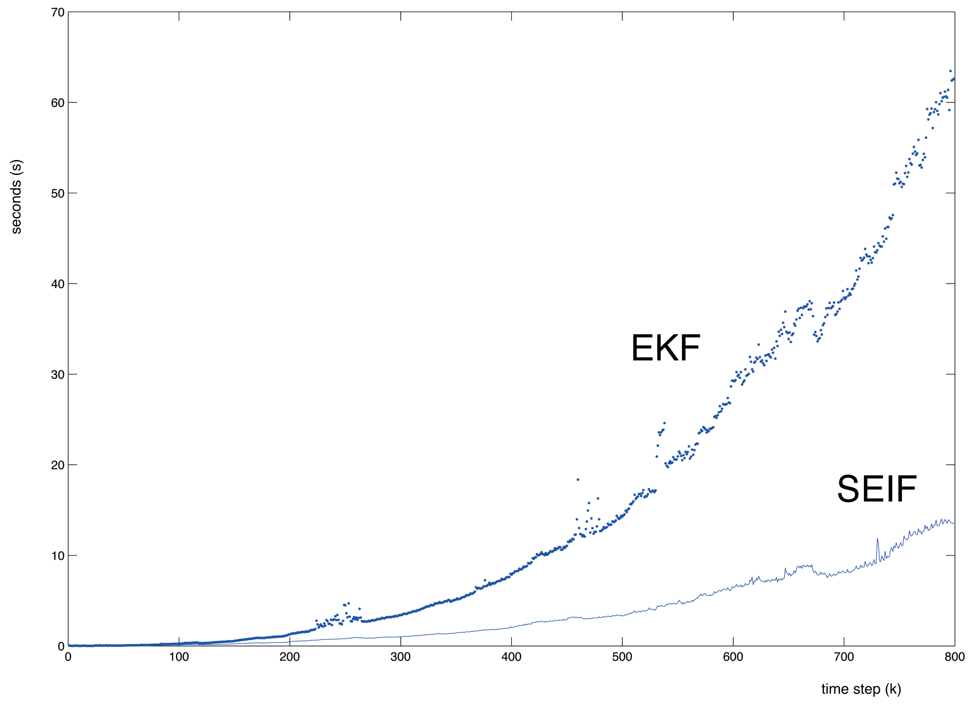

Figure 4.9: EKF vs. ESEIF computation time

4.5 Large Environment Mapping

Having proved that our Omnivideo system is adequate for building a map correctly, and having shown how small loops are correctly closed, a most challenging data set was aquired. Again in an office environment the robot was driven for more 1.5 hours along corridors and offices over a surface of more than 10,000 squared meters. In order to test the ability of our approach to cope with closing large loops, we drove to the starting point after 45 minutes and having already taken 5210 images. The robot was then driven over the same part of the corridors to increase the overlap and finalized at the end of a corridor after having taken 10,325 images in total. After the complete dataset was recorded, the accumulated error in the odometry added up to more than 80 meters in distance and 100 degrees in heading.

We again present two maps of the same trajectory. The first one is based on the odometry readings (see figure 4.10), while the second one represents the SLAM corrected trajectory (see figure 4.11. For this maps we do not show the laser readings as they do not convey additional information.

Figure 4.11 shows the resulting SLAM corrected trajectory of the robot. As we can see, the large loop is correctly closed (all grey circles represent the same spot in the trajectory) even though the accumulated error in odometry was very large. However, the resulting SLAM corrected trajectory is not as truthful as it was for the case of the small dataset. The reason for this is the enormous amount of correction needed to close such a large gap in the loop closing point. When previously seen areas are detected, the whole trajectory needs to be *bent* in order for those areas to overlap. Given the constraints between different robot poses (enforced by both the observations and the odometry), the *bending* is applied over the complete trajectory, reconnecting the loop closing portions, but spreading the rest of the trajectory. This behavior in the loop closing is simmilar of that of a piece of wire that is bent putting both extremes together. As there are forces between the individual cells of the wire, the whole shape of the wire is modified when connecting both extremes. This behavior was also termed "Certainty of Relations despite Uncertainty of Position" by Udo Frese [9] and it is a direct result of the strong relations introduced in the trajectory by both the observations and the odometry.

In figure 4.12 we see the amount of matches found at every time step. As we can see, only in the loop closing points or the stationary moments the amount of observations increases. For the large loop after $5,210$ images, we see a sudden increase in the amount of matches which represents the robot driving the same hallway. As this portion of the trajectory was driven before, the amount of matches doubles after revisiting for the first time. Again, after $7,800$ images, the same hallway is visited again and the number of matches again doubles. The smaller peaks in the plot represent portions where the robot either closed small loops (going around a small room for instance) or the robot stood still for some seconds.

Regarding the computation time of the state recovery, we appreciate in figure 4.13 very interesting results. Initially, and for reasonably large datasets, the Cholmod2 method performs better than the rest. However, this advantage in performance is actually caused by the constant number of observations. Looking closely at figures 4.13 and 4.12, we see how at the moment of the first loop closure (time ste $\approx 5,210$), the growth in computation time of the Cholmod2 increases significantly and in a non linear fashion. This can be clearly seen in figure 4.15 where the computation time divided by the number of non zero entries is displayed. It is clear that the CGS remains approximately constant with the number of observations. This is a very interesting result key to a sucessfull implementation

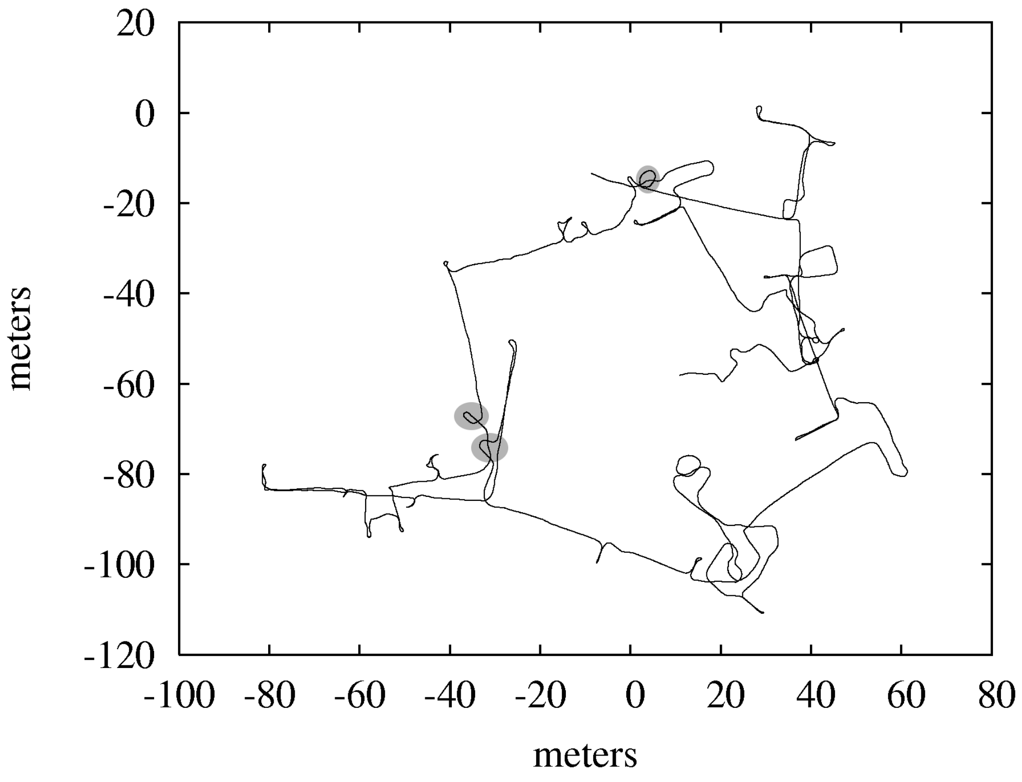

Figure 4.10: Odometry based map - large loop

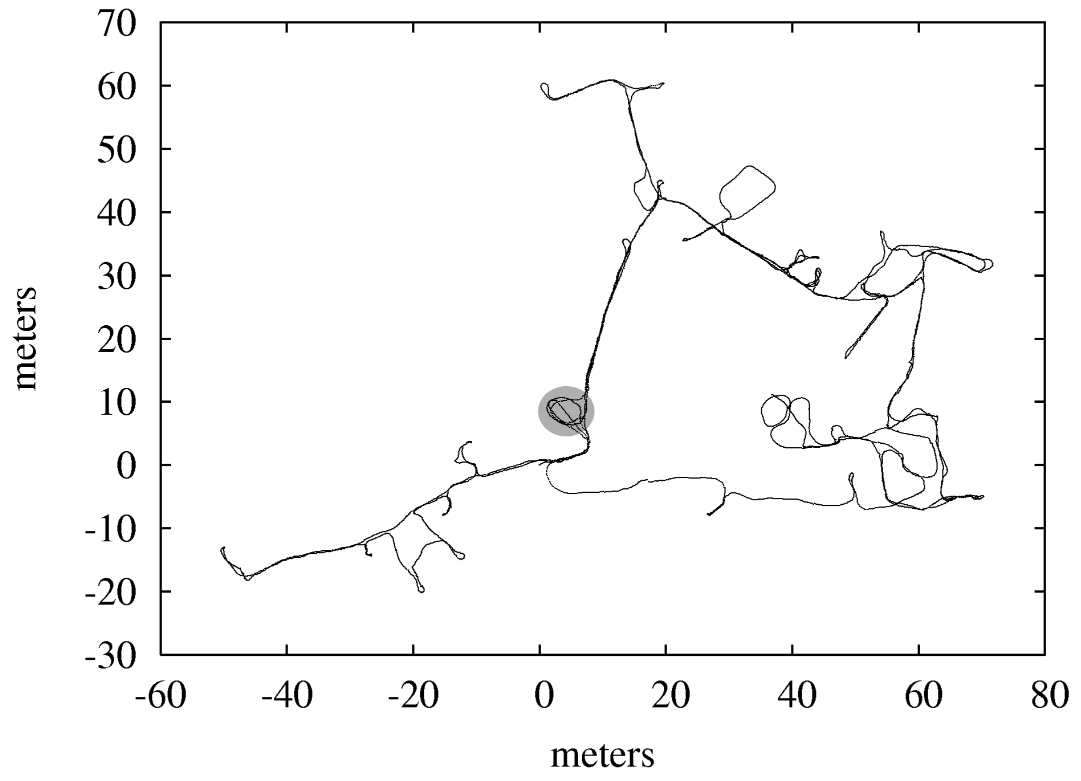

Figure 4.11: SLAM based map - large loop

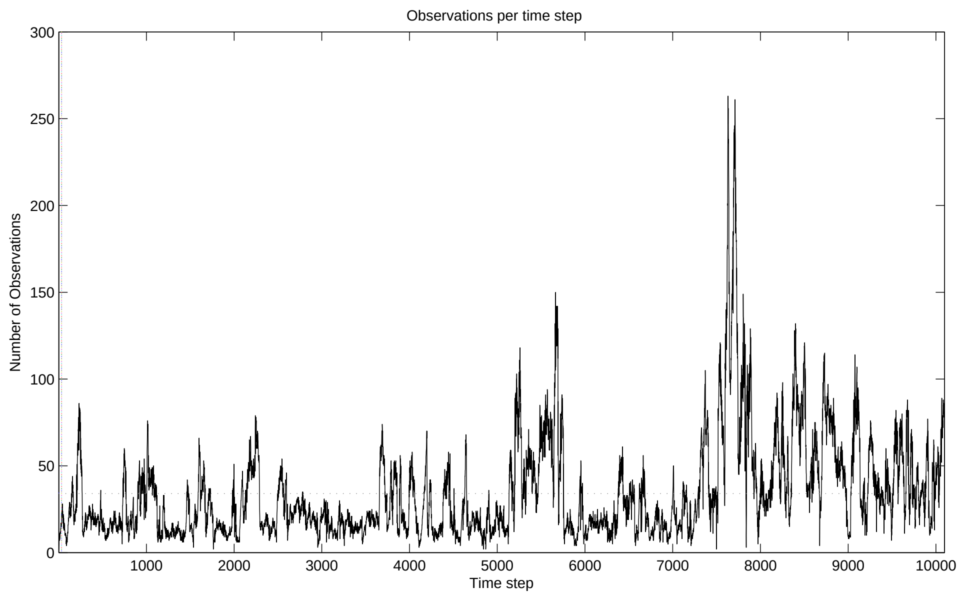

Figure 4.12: Number of observations. Average 34,45.

of the EIF over extremely large datasets. For an efficient implementation, a mixed strategy could be employed, using the Cholmod2 method when the number of observations is limited, and switching to CGS when loops are closed. The naive inversion technique grows so fast that it is barely useful for maps with more than 100 features. The PCG technique shows a resonable performance up to time step $1,000$. The sudden increase in computation time might be caused by an inapropriate conditioning. Lu and CGS behave very similarly though their performance is significantly worst than Cholmod2 for a constant amount of observations. This difference in performance is specially noticeable for large datasets (more than $\approx 2,000$ robot poses). As we can see, the time required by LU or CGS is more than double at time step $3,000$, which implies that the total computation time up to that time step is 27 minutes for LU and CGS and 10 minutes for Cholmod2.

As the CGS is not an exact solution, a comparishon measure was needed to determine wether the CGS was overconfident or conservative in the estimation of the information matrix. Using a measure from [22], we compared the matrices obtained in CGS and Cholmod. The resulting histogram shows how the estimated information matrix in CGS is conservative (values greater than zero) with respect to the exact matrix obtained by Cholmod2.

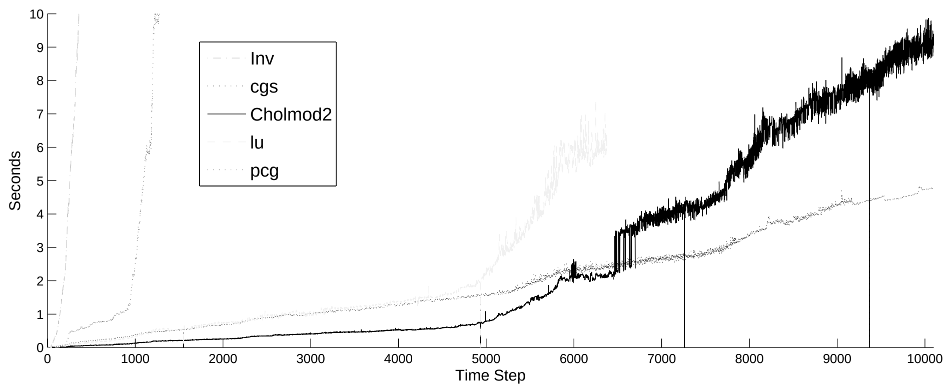

Figure 4.13: Computation speed per time step in state recovery. As we use a trajectory based approach, the time step multiplied by 3 is the number of map features (number of robot poses $\times$ dimensions of each pose)

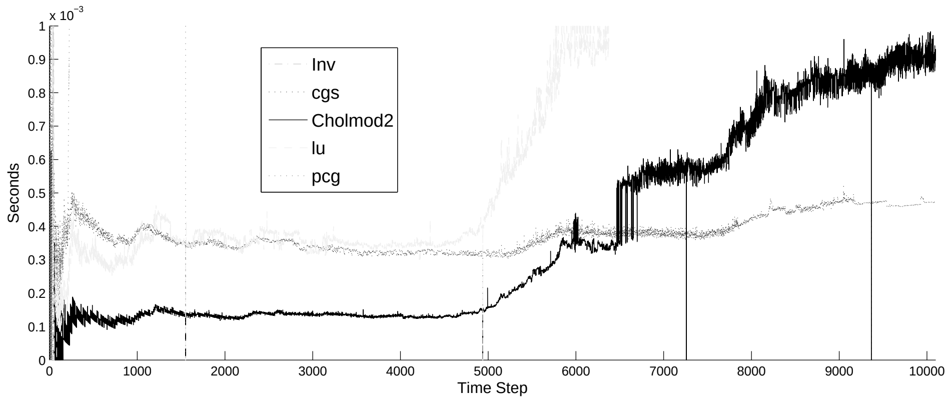

Figure 4.14: Computation speed per time step in state recovery divided by the number of non zero entries in the information matrix.

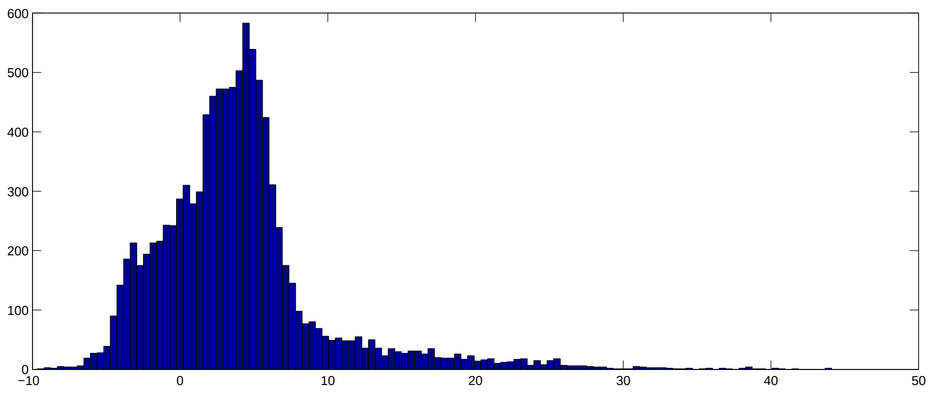

Figure 4.15: Information matrix comparishon between CGS and Cholmod2.

Chapter 5

Conclusions ad Future Work

We have succesfully shown that an omnivideo system is adequate for building an accurate map of indoor office environments even when facing low light conditions. We have shown that estimating the epipolar geometry of two panoramic images and obtaining the relative heading and orientation of each other yileds enough information to create a consistent trajectory map and compensate for the accumulation of error in odometry. Furthermore, by means of a dataset recorded with a mobile robot, we have shown that a very accurate map of a small office environment can be created and maintained by our omnivision trajectory based SEIF approach. Given the computation time required by the ESEIF and the state recovery process, we have also shown that building such a map (for $\approx 1,000$ robot poses) can also be done in real time.

Using a robust image matching algorithm together with the ESEIF, large loops can be effectively closed in extremely large environments. The computational gain of the Information Filter compared with the traditional Extended Kalman Filter solution showed an enourmous improvement in performance. Not only for the small office experiment, but also for the $10,000$ squared meters environment, which is one of the largest experiment on Visual SLAM to our knowledge. Such a large experiment will not be possible with the standard Extended Kalman Filter due to the quadratic time required. Furthermore, it will not be possible without an appropriate state recovery technique, as we have shown that only LU decomposition, CGS and Cholmod2 are sufficiently fast for less than $5,000$ robot poses, though the Cholmod2 is significantly better in terms of global computation time. For extremely large datasets (more than $10,000$ robot poses) we suspect that the CGS will perform better as the computation time growth in our experiments shows a more linear behavior less sensible to number of observations. An efficient implementation will consider using the Cholmod2 for a number of robot poses (map features) below $5,000$ and a constant number of observations. When the number of features increases significantly, for instance when closing large loops, the CGS seems to be the most appropriate choice.

Having observed the amount of error induced by the linearization process by means of an artificial data set our SLAM algorithm could be improved with a more appropriate linearization technique. The essential drawback of the Information Filter is the need to recover the state in order to compute the next filter step. Some approaches regarding partial state recovery could be employed though they also introduce error as they are only an approximation. Another interesting alternative to explore will be the substitution of the non linear function $h(x)$ with an alternative function over the information vector, namely $h^*(\eta)$. This will shortcut the need to recover the state vector x though the definition of such function is difficult to foresee as the information vector lacks geometrical meaning.

Regarding the "bending" process of the trajectory on the loop closing points, we believe, that using a relaxation technique to introduce additional error between certain robot poses, the bending could be enforced over those poses, acting as joints in the bending process. This error introduction could be done by a more detailed odometry error model. By droping the use of a static error covariance R and introducing an improved model that accounts for the additional error when the robot is turning. Such an error model will be integrated in a non-linear motion process.

As an extension of this Thesis, additional experiments will be performed with larger datasets in outdoor environments. Also the planar constraint used in the 3 point algorihtm (epipolar geometry estimation) will be droped and a 3D slam algorithm will be implemented. As part of our goal to develop a Visual SLAM algorithm for extremely large environments we will further investigate all the proposed techniques with the objective to implement a system capable of mapping, navigating and 3D reconstructing a portion of a city.

Appendix A

General Math

A.1 Canonical Representation of a Gaussian

A Gaussian can be expressed using the so called *canonical* representation or *information form*:

$$\phi(x) = exp(g + m^T x - \frac{1}{2} x^T W x) \tag{A.1}$$

where the polynomial coefficients m and W represent the Gaussian and are related to the mean and covariance by:

$$\begin{aligned} W &= \Sigma^{-1} && \text{(A.2)} \\ m &= \Sigma^{-1}\mu && \text{(A.3)} \\ g &= -\frac{1}{2} log \mid 2\pi\Sigma \mid - \frac{1}{2}\mu^T \Sigma^{-1}\mu && \text{(A.4)} \end{aligned}$$

and the Gaussian is usually written as:

$$\phi(x) \equiv \mathcal{N}^{-1}(m, W), \tag{A.5}$$

where the constant term g is omitted.

A.2 Joint Gaussian Distributions

The joint Gaussian distribution for two uncorrelated random variables is defined as:

$$P(z,x) = \mathcal{N}(\left[\begin{array}{c} \mu_z \\ \mu_x \end{array}\right], \left[\begin{array}{cc} \Sigma_{zz} & \Sigma_{zx} \\ \Sigma_{xz} & \Sigma_{xx} \end{array}\right]) \tag{A.6}$$

or in its standard representation:

$$P(z,x) = \frac{1}{2\pi\Sigma} exp\left(-\frac{1}{2}\left(\begin{array}{cc} z-\mu_z & x-\mu_x \end{array}\right)\left(\begin{array}{cc} \Sigma_{zz} & \Sigma_{zx} \\ \Sigma_{xz} & \Sigma_{xx} \end{array}\right)^{-1}\left(\begin{array}{c} z-\mu_z \\ x-\mu_x \end{array}\right)\right) \tag{A.7}$$

Again, expanding the quadratic in the exponential, we can reach a canonical representation of the joint Gaussian:

$$P(z,x) = exp\left(g + \left(\begin{array}{cc} m_z & m_x \end{array}\right)\left(\begin{array}{c} z \\ x \end{array}\right) - \frac{1}{2}\left(\begin{array}{cc} z & x \end{array}\right)\left(\begin{array}{cc} W_{zz} & W_{zx} \\ W_{xz} & W_{xx} \end{array}\right)\left(\begin{array}{c} z \\ x \end{array}\right)\right) \tag{A.8}$$

where:

$$\left(\begin{array}{cc} W_{zz} & W_{zx} \\ W_{xz} & W_{xx} \end{array}\right) = \left(\begin{array}{cc} \Sigma_{zz} & \Sigma_{zx} \\ \Sigma_{xz} & \Sigma_{xx} \end{array}\right)^{-1} \tag{A.9}$$

$$m_z = K_{zz}\mu_z + K_{zx}\mu_x \tag{A.10}$$
$$m_x = K_{xx}\mu_x + K_{xz}\mu_x \tag{A.11}$$

$$g = -\frac{1}{2}log \mid 2\pi\left(\begin{array}{cc} \Sigma_{zz} & \Sigma_{zx} \\ \Sigma_{xz} & \Sigma_{xx} \end{array}\right) \mid -\frac{1}{2}\left(\begin{array}{cc} \mu_z & \mu_x \end{array}\right)\left(\begin{array}{cc} \Sigma_{zz} & \Sigma_{zx} \\ \Sigma_{xz} & \Sigma_{xx} \end{array}\right)^{-1}\left(\begin{array}{c} \mu_z \\ \mu_x \end{array}\right) \tag{A.12}$$

And perhaps another more interesting expression of the inverse of the covariance matrix based on the Inversion Lemma (see further Appendix A.3):

$$\left(\begin{array}{cc} \Sigma_{zz} & \Sigma_{zx} \\ \Sigma_{xz} & \Sigma_{xx} \end{array}\right)^{-1} = \left(\begin{array}{cc} \Sigma_{zz}^{-1} + \Sigma_{zz}^{-1}\Sigma_{zx}(\Sigma_{xx} - \Sigma xz(\Sigma_{zz}^{-1})\Sigma_{zx})^{-1}\Sigma_{xz}\Sigma_{zz}^{-1} & -\Sigma_{zz}^{-1}\Sigma_{zx}(\Sigma_{xx} - \Sigma xz(\Sigma_{zz}^{-1})\Sigma_{zx})^{-1} \\ -(\Sigma_{xx} - \Sigma xz(\Sigma_{zz}^{-1})\Sigma_{zx})^{-1}\Sigma_{xz}\Sigma_{zz}^{-1} & -(\Sigma_{xx} - \Sigma xz(\Sigma_{zz}^{-1})\Sigma_{zx})^{-1} \end{array}\right) \tag{A.13}$$

A.3 The Matrix Inversion Lemma

The so called Matrix Inversion Lemma is a result from the inverse of the nonsingular $n \times n$ partitioned matrix:

$$\left[\begin{array}{cc} P_{11} & P_{12} \\ P_{21} & P_{22} \end{array}\right]^{-1} = \left[\begin{array}{cc} V_{11} & V_{12} \\ V_{21} & V_{22} \end{array}\right] \tag{A.14}$$

where:

$$\begin{aligned}
V_{11} &= P_{11}^{-1} + P_{11}^{-1} P_{12} V_{22} P_{21} P_{11}^{-1} = (P_{11} - P_{12} P_{22}^{-1} P_{21})^{-1} && \text{(A.15)}\\
V_{12} &= -P_{11}^{-1} P_{12} V_{22} = -V_{11} P_{12} P_{22}^{-1} && \text{(A.16)}\\
V_{21} &= -V_{22} P_{21} P_{11}^{-1} = -P_{22}^{-1} P_{21} V_{11} && \text{(A.17)}\\
V_{22} &= P_{22}^{-1} + P_{22}^{-1} P_{21} V_{11} P_{12} P_{22}^{-1} = (P_{22} - P_{21} P_{11}^{-1})^{-1} P_{12} && \text{(A.18)}
\end{aligned}$$

and is defined as:]

$$(A + BCB')^{-1} = A^{-1} - A^{-1} B (B' A^{-1} B + C^{-1})^{-1} B' A^{-1} \tag{A.19}$$

Appendix B

Naming Conventions

- x_k - a posteriori state estimation at time step k
- $\hat{x}_k$ - a priori prediction of state at time step k
- $\check{x}_k$ - ground truth state
- $\hat{z}_k$ - measurment prediction at time step k
- z_k - real measuement at time step k
- R - noise model for the measurements
- Q - noise model for the process
- $\hat{P}_k$ - a priori error covariance
- P_k - a posteriori error covariance
- $\dot{\mathbf{x}}_k^*$ - 3D vector containing the robot pose at time step k
- M - the map or remaining robot poses
- $\mathbf{x}_k$ - position in the X axis at time step k
- $\mathbf{y}_k$ - position in the Y axis at time step k
- θ_k - heading time step k

Bibliography

[1] *Estimation with Applications to Tracking and Navigation.* John Wiley and Sons, Inc., 2001.

[2] A.J.Davidson, Reid, I., Molton, N., and Stasse, O. Mono-slam: Real-time single camera slam. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* (2007).

[3] Booi, O., Terwijn, B., Zivkovic, Z., and B.Kröse. Navigation using an appereance based topological map. In *ICRA* (2007).

[4] Booij, O., Zivkovic, Z., and Kröse, B. Epipolar geometry estimation for mobile robots. submitted.

[5] Brown, R., and Hwang, P. *Introduction to Random Signals and Applied Kalman Filtering, Second Edition.* John Wiley and Sons, Inc., 1992.

[6] Eustice, R., O.Pizarro, and Singh, H. Visually augmented navigation in an unstructured environment using a delayed state history. In *ICRA* (2004).

[7] Eustice, R. M., Singh, H., and Leonard, J. J. Exactly sparse delayed-state filters for view-based slam. *IEEE Transactions on Robotics 22*, 6 (2006).

[8] Folkensson, J., Jensfelt, P., and Christensen, H. Vision slam in the measurement subspace. *Robotics and Automation Magazine* (2005).

[9] Frese, U. A discussion of simultaneous localization and mapping. *Autonomous Robots* (February 2006).

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

[11] Horn, B. Recovering baseline and orientation from essential matrix, January 1990. available from http://ocw.mit.edu/OcwWeb/Electrical-Engineering-and-Computer-Science/6-801Fall-2004/Readings/.

[12] Jacobs, O. *Inotroduction to Control Theory, 2nd Edition.* Oxford University Press, 1993.

[13] Jensfelt, P., kragic, D., Folkensson, J., and Björkman, M. A framework for vision based bearing only 3d slam. *Robotics and Automation Magazine* (2006).

[14] Kröse, B., Booij, O., and Zivkovic, Z. A geometrically constrained image similarity measure for visual mapping, localization and navigation. *European Conference on Mobile Robots* (2007).

[15] Lowe, D. G. Distinctive image features from scale-invariant keypoints. *Int. J. of Computer Vision 60*, 2 (2004), 91–110.

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

[17] Maybeck, P. S. *Stochastic Models, Estimation, and Control, Volume I.* Academic Press, Inc, 1979.

[18] Pfister, S., Roumeliotis, S., and J.W.Burdick. Weighterd line fitting algorithms for mobile robot map building and efficient data representation. *Robotics and Automation Magazine* (2003).

[19] S. Thrun, Y. Liu, D. K., Ng, A., Ghahramani, Z., and Durrant-Whyte, H. Simultaneous localization and mapping with sparse extended information filter. *International Journal of Robotics Research 23(7-8)* (2004).

[20] Schewchuk, J. An introduction to the conjugate gradients method without the agonizing pain. *Technical report, School of Computer Science, Carnegie Mellon University Technical report, School of Computer Science, Carnegie Mellon University Technical report, School of Computer Science, Carnegie Mellon University Technical report, School of Computer Science, Carnegie Mellon University* (1994).

[21] Smith, R., and Cheesman, P. On the representation of spatial undertainty. *Robotics Research, 5(4):5668* (1987).

[22] Walter, M. R., Eustice, R. M., and Leonard, J. J. Exactly sparse extended information filters for feature-based slam. *The International Journal of Robotics Research* (February 2007).

[23] Welch, G., and Bishop, G. *Technical Report, TR 95-041*.

[24] z. Zivkovic, and Booij, O. How did we built our hyperbolic mirror omnidirectional camera - practical issues and basic geometry, technical report. *IAS-UVA-05-4, Informatics Institute, University of Amsterdam* (2006).