TAMPERE UNIVERSITY OF TECHNOLOGY

**Bahram Behzadian**

# Robot Localization with Weak Maps
## Master of Science Thesis

Supervisors: Prof. Dr. Wolfram Burgard
Dr. Luciano Spinello
Dr. Gian Diego Tipaldi
University of Freiburg
Autonomous Intelligent Systems Group

Examiner: Prof. Risto Ritala
Tampere University of Technology
Measurement and Information Technology
The topic approved by the Faculty
of Engineering Sciences on 14.8.2013

# Abstract

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Program in Machine Automation
**Bahram Behzadian : Robot Localization with Weak Maps**
Master of Science Thesis, 52 pages, 0 Appendix pages
5.9.2013
Major: Mechatronics
Examiner: Prof. Risto Ritala
Keywords: Robot Localization, Sketch based map, Unknown map scale, MCL, Scale estimation, Polygonal map, Kalman filter, Geometrical constraints

In this work, we present an approach for indoor localization for a mobile robot based on a weakly-defined prior map. The aim is to estimate the robot's pose where even an incomplete knowledge of the environment is available, and furthermore, improving the information in the prior map according to measurements.

We discuss two different approaches to describe the prior map. In the first approach, a complete map of the environment is given to the robot, but the scale of the map is unknown. The map is represented by occupancy grid mapping. We present a method based on Monte Carlo localization that successfully estimates the robot's pose and the scale of the map.

In the second approach, the prior map is a *2D* sketched map provided by a user, and it does not hold exact metric information of the building. Moreover, some obstacles and features are not fully presented. The aim is to estimate the scale of the map and to modify and correct the prior map knowing the robot's exact pose. The map is represented in the polygonal format in the homogeneous coordinates, and is capable of analyzing the uncertainty of features.

We propose two methods to update prior information in the map. One uses a Kalman filter, and the other is based on Geometrical Constraints. Both methods can partially improve the estimate of the dimensions of rooms and locations and the orientation of walls, but they slightly suffer from data association.

# Acknowledgment

I have taken efforts in this Thesis. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to Prof. Dr. Wolfram Burgard for his guidance and supervision as well as for providing necessary information regarding the project. I would like to express my gratitude towards Prof. Risto Ritala for his kind co-operation and encouragement which help me in the completion of this project.

Furthermore I would like to thank Dr. Luciano Spinello and Dr. Gian Diego Tipaldi for the useful comments, remarks and engagement through the learning process of this master thesis. My thanks and appreciations also go to Nichola Abdo in developing the project and AIS members who have willingly helped me out with their abilities. Finally, I would also like to thank to the TUT International Office for their financial support granted through exchange student scholarship.

# Contents

# Chapter 1

# Introduction

Global robot localization is one of the key problems in the field of mobile robotic systems. It is the problem of determining the robot pose according to a given map, which is important in almost all robotic applications. Most robots do not have precise sensor to determine their exact pose, and the robot pose should be extracted from noisy data, which is typically done in a probabilistic framework like the Bayesian filtering [1].

Several localization algorithms have been suggested in the robotics literature. One can divide these studies to two main groups. The first is based on complete prior knowledge of the environment in the form of a map. Markov localization [2, 3], Extended-Kalman-Filter (EKF) localization [4], Grid localization and Monte Carlo localization [5, 6] are very good examples of probabilistic localization algorithms which require an accurate map as input.

The second group addresses the case when the robot does not have access to a map of the environment and to its exact pose. They are known as Simultaneous Localization and Mapping techniques, or SLAM. The robot is provided with measurements $z_{1:t}$ and controls $u_{1:t}$, and tries to build a map and localize itself in it at the same time. Online SLAM [7] and Full SLAM [8] are the two main forms of the SLAM problem. Online SLAM involves estimating the posterior over the current pose along with the map, whereas, in full SLAM, we try to estimate a posterior over the entire path of robot poses along with the map, instead of just the current pose.

Robot localization is thus a considerable problem whether a complete and exact map of the environment is given to the robot, or the robot has to build a map with sensor measurements and controls information. In this work, we discuss another instance of this problem along this spectrum, which is localization in a map with weak prior information of the environment. We formulate two problems to investigate robot localization with an incompetent knowledge of prior map. The aim is to understand whether this weak information helps to improve localization or causes more uncertainty in the robot pose estimate compared to other scenarios.

## 1.1 Motivation

In many localization scenarios, an exact map of the environment is unavailable. The focus of this work is to consider such scenarios where even an incomplete knowledge of robot environments helps in robot localization. As personal robots gets more ubiquitous, an interactive interface between normal users and a robot for providing environment information becomes more and more relevant. For example, one can draw a hand sketched-map

and the robot to use as a polygonal map. Because humans are not particularly good at producing accurate and detailed maps, the robot has to deal with errors and lack of features.

One primary question that arises at this point is whether noisy data as a prior map will increase the entropy of the problem or improve the localization certainty. The answer could depend on the data association between sensor measurements and the weakly-defined prior map. We believe that a thoughtful data association method causes a significant improvement in localization.

This thesis is presented in two main parts. In the first, we formulate a localization problem where a map of a building floor is available, but the scale of the map is unknown. We try to learn the scale of the map while localizing the robot. In the second part, we present a user-defined sketched map in the form of polygons. We investigate whether the robot would be able to estimate the map scale and correct mistakes in the prior map.

## 1.2 Contribution

Our contribution is divided into three parts:

1. We have designed a localization algorithm based on Monte Carlo Localization which can estimate a pose in a prior map with unknown scaling. Furthermore, it can approximate the scale of the map with acceptable accuracy.

2. We offer a map representation technique in polygonal form that has some novel characteristics. All segments are depicted in a *3-parameter* infinite line format in homogeneous coordinates. The intersection of two consecutive lines determines the corner points. This enables automatic updating of corner points whenever any segment is moved on to a new position. Additionally, all segments acquire individual $3 \times 3$ covariance matrices that represent the likelihood of each segment.

3. We propose two different methods for polygonal map scale estimation and modification. One is based on The Kalman Filter, which is modified for our intention. The second method is based on geometrical constraints. Although these methods offer ways for approximate correction of the map, the final map is by no means a perfect output. We aim to improve this further in future work via the use of other particle filter-based algorithms such as Rao-Blackwellised particle filtering.

## 1.3 Outline

The rest of this thesis is structured as follows. In the next chapter, we discuss related work about different methods in localization and Simultaneous Localization and Mapping. Additionally, we consider some work related to sketch-based maps. In Chapter 3, we briefly review the Particle and Kalman filters. Also, basic principles of Projective Geometry and Geometrical Constraints that are useful to understand this thesis are discussed. In Chapter 4, we introduce a new localization technique in a map with unknown scale, which is based on Monte Carlo Localization. Moreover, it estimates the map scale with agreeable accuracy. In Chapter 5, we deal with different methods for scale estimation and map modification of sketched maps in polygonal format based on hand-drawn

maps such maps lack precise metric information and scale. Chapter 6 describes the results of these experiments, followed by a comparison between different approaches in the previous chapter. Finally, Chapter 7 summarizes what we have achieved in this thesis and discusses how this work can be extended.

# Chapter 2

# Related Work

## 2.1 Localization with Prior Information

Monte Carlo Localization (MCL) is a popular probabilistic method for localization of mobile robots, first proposed in 1999 by Dellaert et al. [5]. Similar to grid-based Markov localization, MCL can be used in both local and global localization settings. MCL has grown to become one of the most common localization algorithms in robotics. The principle of the presented localization approach in this work is based on MCL with the extra function of scale estimation.

Kosecka et al. represented a vision-based crossed localization system based on scale-invariant key-points [9]. In the primary step, topological localization is performed by matching the key-points recognized in the current view, with the database of model views. When the perfect match has been obtained, the corresponding pose between the model view and the current image is rediscovered. They developed this work for topological maps. mainly because they scale better to real-world applications. In contrast to our Localization technique with metrical and deterministic map representations, which can handles the complexity of unstructured environments.

## 2.2 Localization with Weak Prior Information

One method for lidar-based simultaneous localization and mapping that engages aerial images as prior information is proposed by Kuemmerle et al. [10] where the problem of combining data from an aerial map into a robot measurement model using a 3D laser scanner is addressed. It interpolates correspondences tracked down between stereo and three-dimensional range data and the aerial images as constraints into a graph-based SLAM. A notable quantity of clutter in the automatically created landmark maps is obtained. An advanced mode of operations for landmark map generation can help decrease this kind of clutter considerably. Prior information based on aerial images, except those entirely integrated into the SLAM features of this work, is put to work for localization. Single uni-modal position hypotheses are solely thought of as constraints for the graph SLAM algorithm, although the nature of the particle filter is used to generate them.

## 2.3 Localization with Sketched Maps

The problem of localization has been addressed in the context of different types of maps, e.g. feature-based maps, location based maps, occupancy grid maps and graph-like topological maps. Sketch-based maps are usually a representation of environment, by a novice user. Normally, a robot interface is in charge of interaction between the user's knowledge and the robot's prior information. Sketch-based maps are usually used for robot navigation rather than localization. In the following, we present some related work based on sketched maps.

An example of methods that rely on user-provided maps for navigation is the one suggested by Terabayashi, et al. [11]. The system receives a scanned image of a handwritten map, which is a labeled graph describing the topology of the environment, as raw data. Localization is carried out by matching nodes in the graph to a Voronoi diagram obtained from sensor measurements, and which expresses the topological structure of the environment. Their localization process is not designed to handle ambiguous states that are produced by contrast between the map and the real environment. Additionally, using a graph as input critically restricts the expressivity of the map, making it less comprehensible.

Skubic et al. generated a sketch-based interface on a Tablet PC for managing group of robots [12] despite their commitment to the performance of handwritten sketches. This interface controls qualitative input from human users rather than one that has to rely individually on quantitative information.

Numerous projects employ sketch interfaces for robot control. Setalaphruk et al. propose utilization of a scanned, hand-drawn map of an indoor field containing walls and corridors to develop a topological map for controlling a robot [13]. They extended a simulated robot, which applies a geometrically wrong but topologically accurate sketch floor map to cruise. The map is produced in advance by a human.

Parekh et al. presented a paper to discover the object correspondence between a sketched map and the view described by the sketch, represented by an occupancy grid map developed by a robot [14]. A correspondence map between two scene descriptors is created and its confidence assessed. This is a form of reduction in which the scenes are parsed into navigation states by using histograms of force. However, specific kinds of maps that are filled with various redundant features cannot be addressed.

Moreover, Matsuo et al. acquired a stereo-based outdoor localization technique which adopts a hand-drawn line forming building map with high uncertainties [15]. The procedure uses FastSLAM with a particle swarm optimization for map refinement. This method, FastSLAM/PSOM, has been conceived to be practical for outdoor localization for a real mobile robot. Navigation of a mobile robot also needs path planning and obstacle avoidance. It is assumed that a path is also illustrated in a hand-drawn map. However, this approach is not designed for identification of largely-deformed or scale-free maps.

## 2.4 Localization with Polygonal Maps

One of the common methods to fit geometric elements to range data obtained with mobile robots is line extraction. Veeck and Burgard proposed an optimization method for fitting polylines to range data captured by post-processing grid maps [16]. There is a need for a consistent probabilistic prototype over possible maps. A massive quantity of

computational sources are needed that can only be used offline after the data has been collected.

Empirical results for shape comparability applied to range finder data are provided by Latecki et al. [17, 18], where they have employed a position-independent shape similarity measure to match the shape of polygonal curves. The weights are set to take a reasonable perspective between the geometric relations of parallelism, collinearity, and proximity between line segments.

Finally, Movafaghpour et al. implemented different incremental mapping method base on line segments [19]. They utilized a hierarchical structure to exploit a consecutive clustering algorithm to transform data points into point-clusters and finally to line segments. This method incrementally appends new lines to the earlier estimated map lines and integrates them into the overall map.

# Chapter 3

# Fundamentals

In this Chapter, we introduce theoretical background that is needed for our approach in localization and scale estimation technique. First Section is about Particle Filter and Monte Carlo Localization which is the origin of presented scale-invariant Localization method. Section 3.2 is the principle of Kalman filter which is the main key for the first method of polygonal-map modification and correction. Section 3.3 is a brief perception of Projective Geometry and its advantages in map representation. Section 3.4 consists of a common fitting method in statistic. Last section presents a Geometrical constraint which is implemented in the second method of polygonal-map modification.

## 3.1 Monte Carlo Localization

Monte Carlo localization *(MCL)* is a method to find out the location of a robot when given a map of its environment. It is an implementation of the particle filter applied to robot localization. Particle filters are implementations of recursive Bayesian filtering.

Localization methods are used to estimate the posterior of the robot's pose $x_t$ within a known environment m at time t given a series of executed motion commands $u_{1:t-1} = (u_1, ..., u_{t-1})$ and a series of sensor measurements $z_{1:t} := (z_1, ..., z_t)$. The posterior is also called Belief.

$$Bel(x_t) = p(x_t|m, u_{1:t-1}, z_{1:t}) \tag{3.1}$$

In the Bayes Filter method, this posterior is estimated as

$$Bel(x_t) = \eta.p(z_t|x_t) \int p(x_t|u_{t-1}, x_{t-1}) Bel(x_{t-1}) dx_{t-1}, \tag{3.2}$$

where $\eta$ is a normalization factor. $p(z_t|x_t)$ is the probability to measure the observation $z_t$ given that the robot is in pose $x_t$ and is referred to as the sensor model. $p(x_t|u_{t-1}, x_{t-1})$ is the probability that the robot ends up in state $x_t$ if it executes the motion command $u_t$ in state $x_{t-1}$, for this reason this probability distribution is called motion model or action model.

Monte Carlo Localization efficiently represents the posterior of the robot's pose at time $t$ as a set of $n$ weighted particles

$$S_t = \{\langle x_t^{(1)}, w_t^{(1)} \rangle, ..., \langle x_t^{(n)}, w_t^{(n)} \rangle\}, \tag{3.3}$$

where the particle weights $w_t^{(1)}, ..., w_t^{(n)}$ sum up to 1.

The distribution is then approximated by a weighted sum

$$p(x) \simeq \sum_i w^{(i)} \delta_{x^{(i)}}(x). \tag{3.4}$$

where $\delta_x(i)$ is the impulse function centered in $x^{(i)}$. The denser are the samples $x^{(i)}$ in a region, the higher is the probability that the current state falls within that region [20].

Monte Carlo Localization is a Sampling – Importance – Resampling filter as it frequently operates the following three steps:

1. **Sampling:** The algorithm draws a new generation of particles $S_t$ from the previous generation of samples $S_{t-1}$ by drawing from a proposal distribution.

$$q(x_t|m, u_{1:t-1}, z_{1:t}) \tag{3.5}$$

   The optimal proposal distribution would be the target distribution itself.

$$p(x_t|m, u_{1:t-1}, z_{1:t})$$

   As the target distribution is usually not available or not efficiently computable, the motion model $p(x_t|u_{t-1}, x_{t-1})$ is commonly used as the proposal distribution.

2. **Importance weighting:** The samples drawn in step 1 are distributed corresponding to the proposal distribution, which does not equal the target distribution in the overall case. For this reason, the algorithm allocates a weight to each particle as follows:

$$w_t^{(i)} = \frac{p(x_t^{(i)}|m, u_{1:t-1}, z_{1:t})}{q(x_t^{(i)}|m, u_{1:t-1}, z_{1:t})}. \tag{3.6}$$

$$= \alpha . w_{t-1}^{(i)} . \frac{p(z_t^{(i)}|m, x_t^{(i)}) p(x_t^{(i)}|u_{t-1}, x_{t-1}^{(i)})}{q(x_t^{(i)}|m, x_{1:t-1}^{(i)} u_{1:t-1}, z_{1:t})}. \tag{3.7}$$

   where $\alpha$ is a normalization factor arising from Bayes rule, and equal for all particles. If the motion model $p(x_t|u_{t-1}, x_{t-1})$ is used as the proposal distribution, then the computation of the weights turns into

$$w_t^{(i)} = \alpha . w_{t-1}^{(i)} . p(z_t|m, x_t^{(i)}) \tag{3.8}$$

   Therefore the importance weights can by calculated directly according to the observation model

3. **Resampling:** Only a limited amount of particles can be used in a particle filter. As a result, the algorithm resamples the particle set by drawing $n$ particles with replacement in a way that every particle gets drawn with a probability proportional to its importance weight. The particles in the new particle set are distributed corresponding to the target distribution, so the algorithm sets all weights to 1. This principle is illustrated in Figure 3.1.

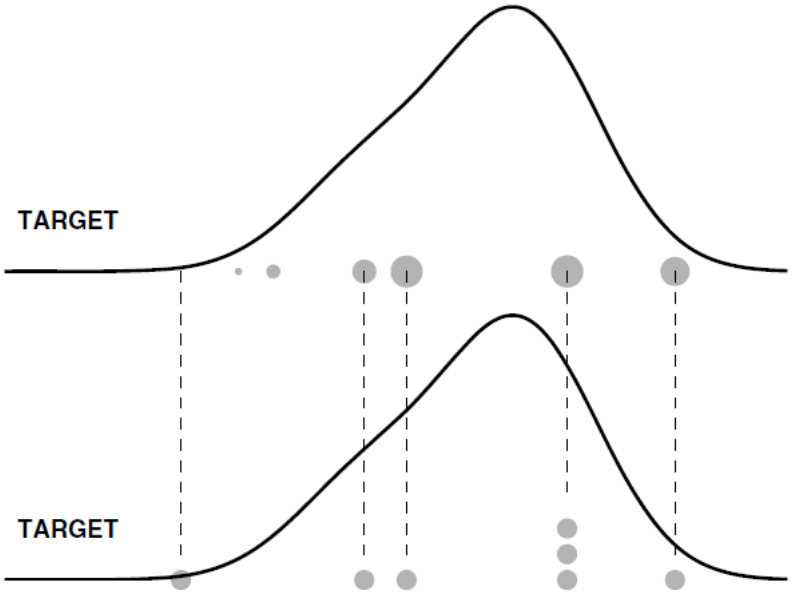



Figure 3.1: Principle of resampling. Top plot: the sample drawn from instrumental distribution with corresponding normalized importance weights pictured by balls with radii equivalent to the normalized weights (the target density is drafted in solid line). Bottom plot: after resampling, all points have the same importance weight, and some have been reproduced[21].

The resampling step draws particles with replacement. Hence, a single particle can be drawn multiple times whereas other particles die out. Therefore, the resampling step decreases the variety of the particle poses, and particles that express the robot's pose well can be wiped out. This difficulty is called particle depletion. For this reason, the resampling step should only be performed when the particle distribution does not cover the posterior distribution well enough, and the re-sampling strategy should maintain the variety of the particles. [1, 22]. In the next Chapter, we introduce Monte Carlo Localization algorithm, modified for unknown map scale.

## 3.2 Kalman Filter

In 1960, R.E. Kalman issued his famous paper defining a recursive answer to the discrete-data linear filtering problem. Later, due in large part to the development of digital computing, the Kalman filter has been the topic of many researches and applications, especially in the field of autonomous or assisted navigation.

The Kalman filters are established on linear dynamic systems involved in the time domain. The filter is extremely powerful in several regards: it carries on estimations of past, present, and future states, and is capable of doing so even when the accurate characteristics of the modeled system is hidden. They are formed on a Markov chain built on linear operators confounded by Gaussian noise. The state of the system is depicted as a vector of real numbers. At each discrete time step, a linear operator is employed to the state to generate the new state, with some additional noise. Then, a further linear operator combined with added noise makes the observed outputs from the hidden state. The Kalman filter can be marked as comparable to the hidden Markov model,

with the essential contrast which the hidden state variables sense values in a continuous space in comparison with the hidden Markov model that take values in a discrete state space. In addition, the hidden Markov model can describe an arbitrary distribution for the next value of the state variables, in opposition to the Gaussian noise model that is employed for the Kalman filter. Sequence information can be obtained from Roweis and Ghahramani [28] and Hamilton [29].

In order to apply the Kalman filter to find out the internal state of a task when only a series of noisy observations is given, the system must be modeled according to the structure of the Kalman filter. This means defining the following matrices:

- $A_t$ the state-transition model
- $B_t$ the control-input model
- $R_t$ the covariance of the process noise
- $C_t$ the observation model
- $Q_t$ the covariance of the observation noise

For each time-step, $t$, as described below

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t \tag{3.9}$$

$\varepsilon_t$ is the process noise which is presumed to be extracted from a zero mean multivariate normal distribution with covariance $R_t$

$$\Longrightarrow \varepsilon_t \sim N(0, R_t)$$

Equation 3.9 explains the state transition probability $p(x_t|u_t, x_{t-1})$. This probability is acquired by filling this state transition equation into the description of the multi-variate normal distribution. The mean of the posterior state is given by $A_t x_{t-1} + B_t u_t$ and the covariance by $R_t$:

$$p(x_t|u_t, x_{t-1}) = \frac{exp\{-\frac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T R_t^{-1}(x_t - A_t x_{t-1} - B_t u_t)\}}{\sqrt[2]{det(2\pi R_t)}} \tag{3.10}$$

At time $t$, an observation $z_t$ of the true state $x_t$ is made corresponding to

$$z_t = C_t x_t + \sigma_t \tag{3.11}$$

where $C_t$ is the measurement model which draws the true state space into the perceived space and $\sigma_t$ is the measurement noise presumed to be zero mean Gaussian noise with covariance $Q_t$.

$$\Longrightarrow \sigma_t \sim N(0, Q_t)$$

The measurement probability is consequently given by the following multivariate normal distribution:

$$p(z_t|x_t) = \frac{exp\{-\frac{1}{2}(z_t - C_t x_t)^T Q_t^{-1}(z_t - C_t x_t)\}}{\sqrt[2]{det(2\pi Q_t)}} \tag{3.12}$$

At the end, the initial belief $bel(x_0)$ must be normal distributed. The mean of this belief is defined as $\mu_0$ and the covariance is defined as $\Sigma_0$

$$bel(x_0) = p(x_0) = \frac{exp\{-\frac{1}{2}(x_0 - \mu_0)^T \Sigma_0^{-1}(x_0 - \mu_0)\}}{\sqrt[2]{det(2\pi\Sigma_0)}} \tag{3.13}$$

These three hypotheses are sufficient to guarantee the posterior $bel(x_t)$ is a Gaussian at any point in time $t$.

The Kalman filter represents the belief $bel(x_t)$ at time $t$ by mean $\mu_t$ and the covariance $\Sigma_t$. The input of Kalman filter is $bel(x_{t-1})$ belief in at time $t-1$ obtained by $\mu_{t-1}$ and $\Sigma_{t-1}$. The predicted belief $\bar{\mu}$ and $\bar{\Sigma}$ is calculated representing the belief $\overline{bel}(x_t)$ one time step later, but before incorporating the measurement $z_t$. This belief is achieved by incorporating the control command $u_t$.

$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \tag{3.14}$$

$$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \tag{3.15}$$

The belief $\overline{bel}(x_t)$ is subsequently transform to $bel(x_t)$ by incorporating the measurement $z_t$.

$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1} \tag{3.16}$$

$$\mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t) \tag{3.17}$$

$$\Sigma_t = (I - K_t C_t)\bar{\Sigma}_t \tag{3.18}$$

The variable $K_t$ is called Kalman gain. It specifies degree to which the measurement is incorporated into the new state estimate [1].

## 3.3 Projective Geometry

This section provides a brief introduction to Projective Geometry and its application in this work. Projective Geometry have a natural application to Computer Graphics; projective geometry is the field of knowledge of geometric characteristics that are invariant under projective transforms. In compare to elementary geometry, projective geometry has a complex projective space, and a particular set of basic geometric concepts. Homogeneous coordinate form a basis for the projective geometry such as Cartesian coordinate in Euclidean geometry. It relates algebra to synthetic projective geometry. Homogeneous coordinates are the key to the quantitative study of projective geometry, a quantitative approach necessary in the study of computer graphics. None of the material presented here is new. We have tried to collect only the necessary information with regards to the presented work [23].

Homogeneous coordinates are widespread in computer graphics because they allow implementation of common operations such as translation, scaling, rotation and perspective projection as matrix operations. They also provided a scale invariant representation for points in the Euclidean plane. This is the main reason for the use of it in this Thesis.

## Terminology

Before introducing the basic theory of Projective Geometry, it is required to explain some terms and compound words, which is commonly noted in this thesis.

$RP^n$ **space:** In mathematics, real projective space, or $RP^n$, is the topological space of lines through 0 in $R^{n+1}$. For instance, $RP^2$ is the real projective plane. It has fundamental applications to geometry since the general form of the real projective plane is as the space of lines passing through the origin in $R^3$.

**Point:** In this work points stand for point-format in *two-dimensional* real projective space, $RP^2$, the reader needs to distinguish them from points in *three-dimensional* space $R^3$.

**Line:** Lines stands for line-format in *two-dimensional* real projective space, $RP^2$, to distinguish them from lines in *three-dimensional* space $R^3$.

**Projective hyperplane:** is a plane located parallel to the $X$ and $Y$ axes in distance $f$ from the $XY$ plane in Cartesian coordinate system $\boldsymbol{R^3}$. Equation $\boldsymbol{Z = 1}$ can present an Projective hyperplane with distance of 1 from origin in $\boldsymbol{R^3}$. The normal vector $\boldsymbol{k} = (0, 0, 1)^T$ is perpendicular to the Projective hyperplanes.

## Representing of geometric objects

One can use homogeneous coordinates for any dimensional spaces. In this project, we focus on Homogeneous coordinates in $2D$. We begin by defining the relevant analytic set in $RP^2$ space. As far as homogeneous coordinates are concerned, $RP^2$ plays a role similar as $R^2$ in coordinating the Euclidean plane.

### Representation of points

Points in homogeneous coordinates are depicted by 3-vector:

$$\boldsymbol{x} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$x$ and $y$ are presenting the coordinates of point with respect to a fixed Cartesian coordinate system in $R^3$. By definition, a 3-vector $\boldsymbol{x}$ pictures a point if and only if the dot product of vector $\boldsymbol{x}$ and $\boldsymbol{k}$ is equal to 1, where $\boldsymbol{k} = (0, 0, 1)^T$ [24]

$$(\boldsymbol{k}, \boldsymbol{x}) = 1$$

### Noise in points

Assume measurement of a point is sensitive to noise. Let $(x, y)$ be the viewed position. If noise $(\Delta x, \Delta y)$ is randomly added, the indicated position is perturbed into $(x + \Delta x, y + \Delta y)$. The noise is supposed to be extremely small and therefore negligible: $|\Delta x| \ll 1, |\Delta y| \ll 1$. In the vector description, the observed value $\boldsymbol{x}$ is randomly perturbed into $\boldsymbol{x} + \boldsymbol{\Delta x}$ by noise $\boldsymbol{\Delta x}$. If $\boldsymbol{\Delta x}$ is noticed as a random variable of mean 0, the uncertainty of the value $\boldsymbol{x}$ is described by the covariance matrix $V[\boldsymbol{x}] = E[\boldsymbol{\Delta x}\boldsymbol{\Delta x}^T]$. Since the noise $\boldsymbol{\Delta x}$ is orthogonal to $\boldsymbol{k}$, it is constrained to be in the *two-dimensional* subspace $\{\boldsymbol{k}\}_L^\perp$ , the set of all vectors orthogonal to $\boldsymbol{k}$. Consequently, the covariance matrix $V[\boldsymbol{x}]$ is

singular and its null space is the *one-dimensional* subspace $\{\boldsymbol{k}\}_L$ created by $\boldsymbol{k}$. The fact that $rank V[\boldsymbol{x}] = 2$ simply states that a point represented by a 3-vector has only ***two*** degrees of freedom. In other words, "**the rank of the covariance matrix indicates the degrees of freedom of the representation.**"

For example if $\Delta x$ and $\Delta y$ are independent Gaussian random variable of mean 0 and variance of $\varepsilon^2$, the covariance matrix of $\boldsymbol{x}$ is given as:

$$V[\boldsymbol{x}] = \begin{pmatrix} \varepsilon^2 & 0 & 0 \\ 0 & \varepsilon^2 & 0 \\ 0 & 0 & 0 \end{pmatrix} = \varepsilon^2 \boldsymbol{P_k} \tag{3.19}$$

where $\boldsymbol{P_k} = \boldsymbol{I} - \boldsymbol{k}\boldsymbol{k}^{\boldsymbol{T}}$ is the projection matrix onto $\boldsymbol{XY}$ plane [24].

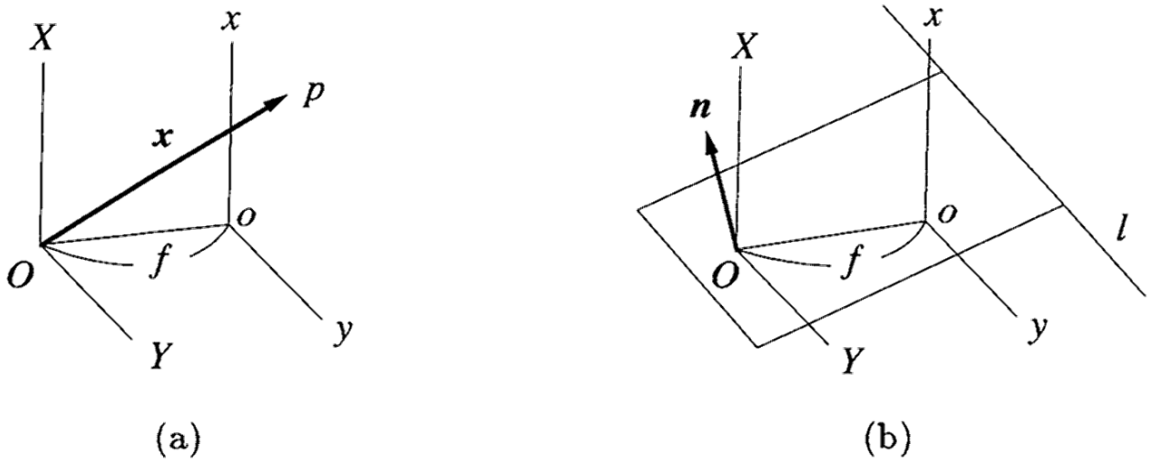



Figure 3.2: (a) representation of point. (b) Representation of a line [24].

**Representation of lines**

A line $l$ is expressed by its mathematical statement:

$$Ax + By + C = 0 \tag{3.20}$$

Since the coefficients $A, B$, and $C$ are defined entirely up to scale, we require the normalization $A^2 + B^2 + C^2 = 1$. Then, a line $l$ is depicted by a unit 3-vector :

$$\boldsymbol{n} = \begin{pmatrix} A \\ B \\ C \end{pmatrix} \tag{3.21}$$

This description is not unique: $\boldsymbol{n}$ and $-\boldsymbol{n}$ maintain the equal line. If the Projective hyperplane is observed as a *two-dimensional* projective space, the three elements of the 3-vector $\boldsymbol{n}$ can be translated as the homogeneous coordinates of the line it expresses. Since A and B cannot be zero at the same time, an arbitrary 3-vector $\boldsymbol{n}$ denotes a line if and only if

$$\|\boldsymbol{n}\| = 1, \qquad\qquad \boldsymbol{n} \neq \pm\boldsymbol{k}$$

The distance of the line $\boldsymbol{n}$ depicted by $d$ from the origin $\boldsymbol{O}$ is:

$$d = \frac{|C|}{\sqrt{A^2 + B^2}} = \frac{|(\boldsymbol{k}, \boldsymbol{n})|}{\sqrt{1 - (\boldsymbol{k}, \boldsymbol{n})^2}} \tag{3.22}$$

The lines of view of all the points on a line $l$ determine a space plane. The vector representation can be thought of as recognizing the line $l$ with the unit surface normal $\boldsymbol{n}$ to that space plane. In fact, the space plane defined by a line $Ax + By + C = 0$ is $AX + BY + CZ = 0$, which has surface normal $\boldsymbol{n} = (A, B, C)^T$ (Fig 3.2b).

### Noise in lines

A degree of uncertainty is associated with measurement of a line. Assume a line expressed by $\boldsymbol{n}$ is randomly perturbed into a position denoted by $\boldsymbol{n} = \boldsymbol{n} + \boldsymbol{\Delta n}$ . If the error $\boldsymbol{\Delta n}$ is noticed as a random variable of mean $\boldsymbol{0}$, the uncertainty of the value $\boldsymbol{n}$ is distinguished by the covariance matrix $\boldsymbol{V}[\boldsymbol{n}] = \boldsymbol{E}[\boldsymbol{\Delta n} \boldsymbol{\Delta n}^{\boldsymbol{T}}]$. Since $\boldsymbol{n}$ is normalized into a unit vector, the error $\boldsymbol{\Delta n}$ is orthogonal to $\boldsymbol{n}$ to a first approximation. As a result, the covariance matrix $\boldsymbol{V}[\boldsymbol{n}]$ is singular and its null space is $\{\boldsymbol{n}\}_{\boldsymbol{L}}$. Hence, even though a line is depicted by a 3-vector $\boldsymbol{n}$, it has two degrees of freedom: rank$\boldsymbol{V}[\boldsymbol{n}] = 2$ [24].

## Distance between a point and a line

Point $p$ and line $l$ are incident to each other if $p$ is on $l$, or $l$ passes through $p$ (Fig 3.3a). Let $(x, y)$ be the coordinates of $p$. If line $l$ is realized by $A, B$, and $C$, point $p$ and line $l$ are incident to each other if and only if $Ax + By + C = O$. In other words, a point $p$ depicted by $\boldsymbol{x}$ and a line $l$ expressed by $\boldsymbol{n}$ are incident to each other if and only if the dot product of $\boldsymbol{n}$ and $\boldsymbol{x}$ is equal to 0.

$$(\boldsymbol{n}, \boldsymbol{x}) = 0$$

If $\boldsymbol{x}$ is considered as a variable, the equation $(\boldsymbol{n}, \boldsymbol{x}) = 0$ stands for equation of line[24]. The distance $D(p, l)$ between a point $p$ expressed by $\boldsymbol{x}$ and a line $l$ expressed by $\boldsymbol{n}$ is

$$D(p, l) = \frac{|(\boldsymbol{n}, \boldsymbol{x})|}{\sqrt{1 - (\boldsymbol{k}, \boldsymbol{n})^2}} \tag{3.23}$$

## Computing the point of intersection of two lines

Two lines $(\boldsymbol{n_1}, \boldsymbol{x}) = 0$ and $(\boldsymbol{n_2}, \boldsymbol{x}) = 0$ are parallel to each other if and only if $|\boldsymbol{n_1}, \boldsymbol{n_2}, \boldsymbol{k}| = 0$ (Fig 3.3b). If lines $(\boldsymbol{n_1}, \boldsymbol{x}) = 0$ and $(\boldsymbol{n_2}, \boldsymbol{x}) = 0$ are not parallel, they intersect at a single point $\boldsymbol{x}$ that is obtained from the following equation :

$$\boldsymbol{x} = \frac{\boldsymbol{n_1} \times \boldsymbol{n_2}}{|\boldsymbol{n_1}, \boldsymbol{n_2}, \boldsymbol{k}|} \tag{3.24}$$

## Computing of the line passing through two points

A line $(\boldsymbol{n}, \boldsymbol{x}) = 0$ that passes through two different points $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ is named the joining line of $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$. Since $(\boldsymbol{n}, \boldsymbol{x_1}) = 0$ and $(\boldsymbol{n}, \boldsymbol{x_2}) = 0$, the vector $\boldsymbol{n}$ must be orthogonal

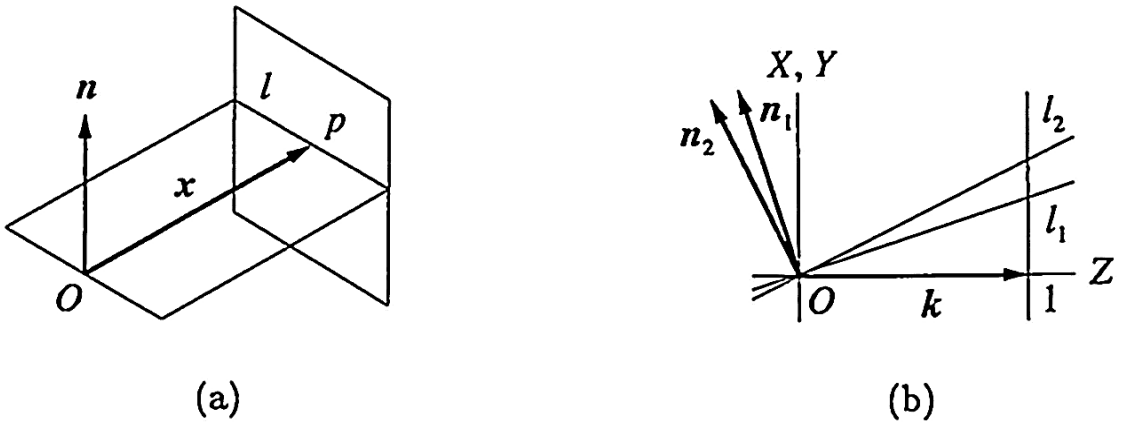



Figure 3.3: (a) Incidence of point $p$ and line $l$. (b) Parallel lines $l_1$ and $l_2$. [24].

to the pair $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ (Fig 3.4b). Also, n is normalized into a unit vector (Fig 3.4b). Therefore the equation below defines a line as long as $\boldsymbol{x_1} \neq \boldsymbol{x_2}$:

$$\boldsymbol{n} = \pm \boldsymbol{N}[\boldsymbol{x_1} \times \boldsymbol{x_2}] \tag{3.25}$$

Where $\boldsymbol{N}$ stands for line normalizer.

### Covariance matrix of the line passing through two points

Assume $\boldsymbol{V}[\boldsymbol{x_1}]$ and $\boldsymbol{V}[\boldsymbol{x_2}]$ are the covariance matrices of points $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$. If $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are perturbed into $\boldsymbol{x_1} + \boldsymbol{\Delta x_1}$ and $\boldsymbol{x_2} + \boldsymbol{\Delta x_2}$, the vector $\boldsymbol{n}$ is perturbed into $\boldsymbol{n} + \boldsymbol{\Delta n}$ accordingly. The error to a first approximation is:

$$\boldsymbol{\Delta n} = \pm \frac{\boldsymbol{P_n}(\boldsymbol{\Delta x_1} \times \boldsymbol{x_2} + \boldsymbol{x_1} \times \boldsymbol{\Delta x_2})}{\|\boldsymbol{x_1} \times \boldsymbol{x_2}\|} \tag{3.26}$$

Where $\boldsymbol{P_n} = \boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^{\boldsymbol{T}}$ is the projection matrix along $\boldsymbol{n}$. If $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are statistically independent, the covariance matrix $\boldsymbol{V}[\boldsymbol{n}] = \boldsymbol{E}[\boldsymbol{\Delta n}\boldsymbol{\Delta n}^{\boldsymbol{T}}]$ of $\boldsymbol{n}$ is presented by: [24]

$$\boldsymbol{V}[\boldsymbol{n}] = \frac{\boldsymbol{P_n}(\boldsymbol{x_1} \times \boldsymbol{V}[\boldsymbol{x_2}] \times \boldsymbol{x_1} + \boldsymbol{x_2} \times \boldsymbol{V}[\boldsymbol{x_1}] \times \boldsymbol{x_2})\boldsymbol{P_n}}{\|\boldsymbol{x_1} \times \boldsymbol{x_2}\|^2} \tag{3.27}$$

For example, if each coordinate is perturbed by Gaussian noise independently of mean 0 and variance $\epsilon^2$, the covariance matrices of $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are $\boldsymbol{V}[\boldsymbol{x_1}] = \boldsymbol{V}[\boldsymbol{x_2}] = \boldsymbol{\epsilon^2 P_k}$ . Then we have :

$$\boldsymbol{x_i} \times \boldsymbol{P_k} \times \boldsymbol{x_i} = \boldsymbol{x_i} \times (\boldsymbol{I} - \boldsymbol{k}\boldsymbol{k}^{\boldsymbol{T}}) \times \boldsymbol{x_i} \tag{3.28}$$

$$= \|\boldsymbol{x_i}\|^{\boldsymbol{2}}\boldsymbol{I} - \boldsymbol{x_i}\boldsymbol{x_i^T} - (\boldsymbol{x_i} \times \boldsymbol{k})(\boldsymbol{x_i} \times \boldsymbol{k})^{\boldsymbol{T}} \tag{3.29}$$

## 3.4 Line Fitting and Total Least Squares

In this Section, we explore a method for fitting a geometric line to various geometric points in an optimal manner in the presence of noise. First, we describe some statistical term such as Least Squares Fitting and Total Least Squares.

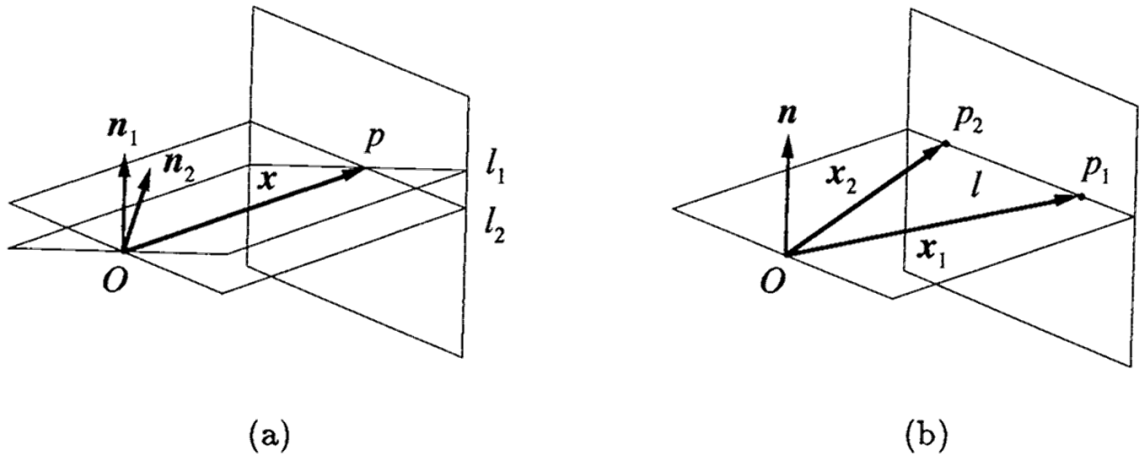



Figure 3.4: (a) The intersection $p$ of line $l_1$ and $l_2$. (b) The join $l$ of points $p_1$ and $p_2$. [24].

Least Squares Fitting (LS) is a statistical method for detecting the best-fitting curve or line to a given set of points by minimizing the sum of the squares of the residuals of the points from the curve. The sum of the squares of the residuals is preferable instead of their absolute values since this bring in the residuals to be used as a continuous differentiable quantity. However, outlying points can have an excessive impact on the fit. Because, the squares of residuals are applied. This is an undesirable characteristic and may not be acceptable depending on the problem.

Total least squares (TLS) is a theory that precedes a technique that has been individually developed in different literature. It has been known by different names, for example it is known as the "errors-in-variables" model in statistical literature. For a linear case, the technique needs computationally fast methods for reaching the numerical solution. With the approach of modern computer technology, it is now possible to calculate the solution to problems including wide numbers of variables and measurements. These numerical methods have taken a highly advanced level in the past few decades.[25]

**Principles** The problem presented is of finding a vector $\beta$ such that $X\beta = y$ when the data matrix $X$ and the observation vector $y$ are given. When there are more equations than unknowns, the set is over-determined with no precise answer. Accordingly, the equation is signified by $X\beta \approx y$. The model holds equations which are linear in the parameters arising in the parameter vector $\beta$, so the residuals are given by

$$\boldsymbol{r = y - X\beta} \tag{3.30}$$

An optimal estimate of $\beta$ can be obtained by the optimization:

$$\min \|X\beta - y\|_2, \qquad X \in R^{m\times n}, y \in R^m \tag{3.31}$$

Any minimizing $\beta$ is called a (linear) least squares (LS) solution of the set $X\beta \approx y$.

There are $m$ measurements in $y$ and $n$ parameters in $\beta$ with $m > n$. $X$ is a $m \times n$ matrix whose elements are functions of the independent variables $x$. The weight matrix $W$ is the inverse of the covariance matrix of the measurements $y$. The independent variables

are assumed to be *error-free*. The parameter estimations are detected by placing the gradient equations to zero, which ends in the normal equations [26]

$$\mathbf{X^T W X \boldsymbol{\beta} = X^T W y} \tag{3.32}$$

In Total Least Squares, $x$ and $y$ are both subject to error, with covariance matrices $\mathbf{V}_x$ and $\mathbf{V}_y$ respectively. In this case the objective function can be addressed as:

$$S = \mathbf{r_x^T V_x^{-1} r_x + r_y^T V_y^{-1} r_y} \tag{3.33}$$

where $\mathbf{r}_x$ and $\mathbf{r}_y$ are the residuals in $x$ and $y$. In comparison, the objective function, S, in the least squares method is:

$$S = \mathbf{r^T W r} \tag{3.34}$$

where $W$ is a weighting matrix. The residuals $\mathbf{r}_x$ and $\mathbf{r}_y$ cannot be independent of each other. They must be constrained by some kind of relationship. These constraints are expressed by $m$ condition equations. This relationship can be expressed as a model function $\mathbf{f(r_x, r_y, \boldsymbol{\beta})}$

$$\mathbf{F = \Delta y - \frac{\partial f}{\partial r_x} r_x - \frac{\partial f}{\partial r_y} r_y - X \Delta \boldsymbol{\beta} = 0} \tag{3.35}$$

The aim is to minimize the objective function subject to the constraints. It is solvable by employing of Lagrange multipliers. After a few algebraic manipulations, the equation below is acquired:

$$\mathbf{X^T M^{-1} X \Delta \boldsymbol{\beta} = X^T M^{-1} \Delta y} \tag{3.36}$$

where $M$ is the covariance matrix corresponding to variables, and $\Delta \boldsymbol{y}$ and $\Delta \boldsymbol{\beta}$ represent the measurement errors [26].

$$\mathbf{M = K_x V_x K_x^T + K_y V_y K_y^T;\ K_x = -\frac{\partial f}{\partial r_x},\ K_y = -\frac{\partial f}{\partial r_y}} \tag{3.37}$$

**Deming Regression**

Deming regression, named after W. Edwards Deming, is an *errors-in-variables* paradigm which attempts to get the line of best fit for a *two-dimensional* data set. It considers errors in measurements on both the *x*- and the *y-axis*. It is a particular case of total least squares, which accepts any number of predictors and suggests a more complex error formation. Deming regression is equal to the maximum likelihood estimation of an *errors-in-variables* model. The errors for the two variables are presumed to be independent with a normal distribution. The ratio of their variances indicated by $\delta$ [26]. Suppose that the accessible data $(y_i, x_i)$ are incorrect estimated measurements of the precise values $(y_i^*, x_i^*)$:

$$y_i = y_i^* + \varepsilon_i \tag{3.38}$$

$$x_i = x_i^* + \eta_i \tag{3.39}$$

where errors $\varepsilon$ and $\eta$ are independent, and the ratio of variances is presumed to be:

$$\delta = \frac{\sigma_\varepsilon^2}{\sigma_\eta^2}. \tag{3.40}$$

In general practice, the variance of the $x$ and $y$ parameters is usually unexplained, but where the observation system for $x$ and $y$ is the same they seem to be the same. Accordingly, in this case we have $\delta = 1$ . To find the line of best fit, the weighted sum of squared residuals (SSR) of the model must be minimized [27].

$$\min_{\beta_0,\beta_1,x_1^*,\ldots,x_n^*} SSR = \sum_{i=1}^{n} \left( \frac{\varepsilon_i^2}{\sigma_\varepsilon^2} + \frac{\eta_i^2}{\sigma_\eta^2} \right) = \frac{1}{\sigma_\varepsilon^2} \sum_{i=1}^{n} \left( (y_i - \beta_0 - \beta_1 x_i^*)^2 + \delta (x_i - x_i^*)^2 \right) \quad (3.41)$$

**Fitting a line to points**

Assume we are given $N$ points $\boldsymbol{x_\alpha}, \alpha = 1, \ldots, N$, and we have the task of fitting a line $\boldsymbol{(n, x) = 0}$ to them. Then we have :

$$\boldsymbol{x_\alpha = \overline{x}_\alpha + \Delta x_\alpha}$$

where $\boldsymbol{\Delta x_\alpha}$ stands for independent random variable with mean 0 and covariance matrix of $\boldsymbol{V[x_\alpha]}$. We assume $\boldsymbol{(n, \overline{x}) = 0}$ which has rank 1. An optimal evaluation of $\boldsymbol{n}$ can be acquired by the optimization

$$min\ \boldsymbol{J[n]} = \sum_{\alpha=1}^{N} \frac{\boldsymbol{(n, x_\alpha)^2}}{\boldsymbol{(n, V[x_\alpha] n)}} \quad (3.42)$$

under the constraint $\|\boldsymbol{n}\| = 1$. The covariance matrix of the solution $\hat{\boldsymbol{n}}$ is

$$V[\hat{n}] = \left( \sum_{\alpha=1}^{N} \frac{\boldsymbol{(P_{\hat{n}} \hat{x}_\alpha)(P_{\hat{n}} \hat{x}_\alpha)^T}}{\boldsymbol{(\hat{n}, V[x_\alpha] \hat{n})}} \right)^- \quad (3.43)$$

where $\boldsymbol{P_{\hat{n}}}$ is the projection matrix along $\hat{\boldsymbol{n}}$, and $\hat{\boldsymbol{x}}_\alpha$ is the optimally corrected value of $\boldsymbol{x_\alpha}$. The rank of $\boldsymbol{V[\hat{n}]}$ is 2. In the case that each coordinate is perturbed independently by Gaussian noise of mean 0 and variance $\epsilon^2$, point $\boldsymbol{x_\alpha}$ has covariance matrix $\boldsymbol{V[x_\alpha]} = \epsilon^2 \boldsymbol{P_k}$, so equation 3.42 reduces to the least-squares optimization

$$min\ \boldsymbol{J_0[n]} = \sum_{\alpha=1}^{N} \frac{\boldsymbol{(n, x_\alpha)^2}}{\boldsymbol{1 - (k, n)^2}} = \sum_{\alpha=1}^{N} D(p_\alpha, l)^2 \quad (3.44)$$

$D(p_\alpha, l)^2$ stands for distance between $\alpha th$ point $p_\alpha$ and line $l$ fitted to the data. (see fig 3.5)

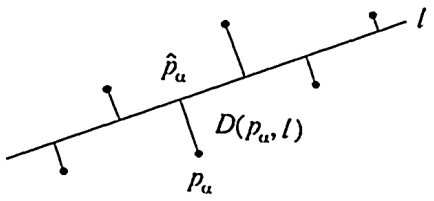



Figure 3.5: Line fitting by least squares [24].

The covariance matrix of the solution $\hat{\boldsymbol{n}}$ for points $\{(x_\alpha, y_\alpha)\}$ is obtained by

$$V[\hat{n}] = \frac{\epsilon^2}{1+\hat{d}^2} \begin{pmatrix} \sum_{\alpha=1}^{N} \hat{x}_\alpha^2 & \sum_{\alpha=1}^{N} \hat{x}_\alpha \hat{y}_\alpha & \sum_{\alpha=1}^{N} \hat{x}_\alpha \\ \sum_{\alpha=1}^{N} \hat{x}_\alpha \hat{y}_\alpha & \sum_{\alpha=1}^{N} \hat{y}_\alpha^2 & \sum_{\alpha=1}^{N} \hat{y}_\alpha \\ \sum_{\alpha=1}^{N} \hat{x}_\alpha & \sum_{\alpha=1}^{N} \hat{y}_\alpha & N \end{pmatrix}^{-} \tag{3.45}$$

where $\hat{d}$ is the distance of the line $(\hat{\boldsymbol{n}}, \boldsymbol{x}) = 0$ from the origin $\boldsymbol{O}$ (see eq 3.22). Here, $(\hat{\boldsymbol{x}}_{\boldsymbol{\alpha}}, \hat{\boldsymbol{y}}_{\boldsymbol{\alpha}})$ is the optimally corrected position of $(\boldsymbol{x}_{\boldsymbol{\alpha}}, \boldsymbol{y}_{\boldsymbol{\alpha}})$ in our scenario obtained by orthogonal projection onto the fitted line $\hat{\boldsymbol{n}}$, consequently we have

$$\hat{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{\Delta x} \tag{3.46}$$

$$\boldsymbol{\Delta x} = -\frac{(\boldsymbol{n}, \boldsymbol{x})\boldsymbol{P_k n}}{\boldsymbol{1 - (k, n)^2}} \tag{3.47}$$

## 3.5 Geometrical Constraints

Multiple geometric objects corresponding to a constraint may not satisfy it, if each object is independently perceived in the attendance of noise. This section introduces a statistically optimal technique to correct and adjust the positions of geometric objects so that they satisfy a necessary constraint. The basic principle is the minimization of the *Mahalanobis distance* explained in terminologies of the covariance matrices of the objects. Although, Kanatani has represented the principle formulation of geometric correction in his book[24], in this section we exclusively clarify terms which are related to ongoing geometric problems in two dimensional space.

### Simultaneous correction for incidence of point and line

Assume point $\boldsymbol{x}$ and line $(\boldsymbol{n}, \boldsymbol{x}) = \boldsymbol{0}$ are, sequentially, estimations of a point $p$ and a line 1 that should be incident to each other in the *non-existence* of noise. Think of the problem of optimally correcting them so as to force them incident. In different words, we desire to obtain $\boldsymbol{\Delta x}$ and $\boldsymbol{\Delta n}$ such that $\bar{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{\Delta x}$ and $\bar{\boldsymbol{n}} = \boldsymbol{n} - \boldsymbol{\Delta n}$ satisfy

$$(\bar{\boldsymbol{n}}.\bar{\boldsymbol{x}}) = \boldsymbol{0} \tag{3.48}$$

The rank of the constraint is 1. Allow $V[x]$ and $V[n]$ be the a priori covariance matrices of $\boldsymbol{x}$ and $\boldsymbol{n}$, sequentially. If the point and the line are statistically independent, the problem can be formulated as the optimization [24]

$$min\ J = (\Delta x, \bar{V}[x]^{-}\Delta x) + (\Delta n, \bar{V}[n]^{-}\Delta n) \tag{3.49}$$

under the linearised constraint

$$(\bar{n}, \Delta x) + (\bar{x}, \Delta n) = (n, x) \tag{3.50}$$

$$\Delta x \in \{k\}_L^{\perp}, \quad \Delta n \in \{\bar{n}\}_L^{\perp} \tag{3.51}$$

$$\tag{3.52}$$

The first order solution is given by

$$\Delta x = \frac{(n, x)V[x]n}{(n, V[x]n) + (x, V[n]x)} \tag{3.53}$$

$$\Delta n = \frac{(n,x)V[n]x}{(n,V[x]n)+(x,V[n]x)} \tag{3.54}$$

A realistic form of the correction is

$$\hat{x} = x - \Delta x, \qquad\qquad \hat{n} = N[n - \Delta n]$$

The posterioriti covarince matrices of the corrected values $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{n}}$ are

$$V[\hat{x}] = V[x] - \frac{(V[x]\hat{n})(V[x]\hat{n})^T}{(\hat{n},V[x]\hat{n})+(\hat{x},\hat{V}[n]\hat{x})} \tag{3.55}$$

$$V[\hat{n}] = \hat{V}[n] - \frac{(\hat{V}[n]\hat{x})(\hat{V}[n]\hat{x})^T}{(\hat{n},V[x]\hat{n})+(\hat{x},\hat{V}[n]\hat{x})} \tag{3.56}$$

$$\tag{3.57}$$

where

$$\hat{V}[n] = P_{\hat{n}} V[n] P_{\hat{n}}.$$

# Chapter 4

# Global Localization and Map Scale Estimation

In this Chapter, we discuss the first approach towards localization in the weakly-defined prior map. The problem arises from localization in a map with unknown scale. Assume we are given a map that is made by the grid occupancy mapping technique. The map's characteristics are consistent, but the scale of the map is not available. The goal is to estimate the position of the robot and the map-scale at the same time when the information from laser range finder and control commands are available.

## 4.1 Approach

Several methods have been introduced for global localization in mobile robotics, and the most popular one is Monte Carlo Localization, which is based on probabilistic theory. References to articles on MCL are given in the *related work*. In Chapter 3 *Fundamentals*, a summary of principles of MCL can be found. For more detailed information, *Probabilistic Robotics* by Thrun, Burgard and Fox is an excellent reference [1].

Basically Monte Carlo localization has four main steps. In the following, we briefly compare the difference between our technique and MCL step by step.

### Step 1: Set of Weighted Particles

The basic MCL algorithm represents the belief $bel(x_t)$ of Robot about its position in the map with a set of $M$ particles $\chi_t = \{x_t^{[1]}, x_t^{[2]}, ..., x_t^{[n]}\}$. These particles are distributed uniformly over the state space. They are indicated with three parameters of $x, y, \theta$.

$$\boldsymbol{x_t} = (x, y, \theta)$$

In this problem, the unknown global scale of the map necessitates an addition of a factor to the pose such that each particle can separately evaluate its belief about size of the environment. Therefore, we added another parameter to the robot pose and denote it with the **scale**.

$$\boldsymbol{x_t} = (x, y, \theta, scale)$$

The scale factor shows the robot's belief about the size of the environment. As given map is in the form of occupancy grid, the scale is defined as $(meter/pixel)$. Map in pixel is another form of map representation of the occupancy grid map, in this work. The scale

factor is drawn from a uniform distribution in the interval from 1 centimeter per pixel to 1 meter per pixel.

$$scalefactor \sim \mathcal{U}(0.01, 1) \tag{4.1}$$

In this step, the weight of particles which represent the probability of existence of each particle, is equal. We assign uniform importance to each particle.

## Step 2: Motion model

Within a particle filter, the samples generated at each time step deal with some provided proposal distribution. A common application for mobile robots is taking the probabilistic motion model exactly as this proposal. The robot Motion model is one of the main steps in all filter algorithm methods used in mobile robotics. Odometry motion model and Velocity motion model are two of popular models in Robot Motion. The decision to implement either model over another is entirely arbitrary and discretionary.

Odometry motion model is the chosen model for the purpose of this work as it offers a greater degree of precision. Additionally, Odometry motion model is more user friendly in compare to the Velocity motion model. Both models have deficiencies in drift and slippage. Although, contrasted with Odometry model, practically, Velocity model cannot offer 100% compliance between the mathematical model and the motion control model. In other words, Velocity motion model suffers from the mismatch between the actual motion controllers and its crude mathematical model[1]. In Odometry model, information is acquired by applying a wheel encoder in mobile robots. The algorithm sample motion model odometry is the one we found appropriate for presenting work. However, we modified the sample motion model for this work, and in the Section 4.2 explained the details.

## Step 3: Measurement model

The purpose of the measurement model in MCL is to evaluate the probability of a hypothetical pose of the robot, given a map of the environment and sensor measurements. It is appealing to know the likelihood of a robot being at a particular pose given what the robot is "seeing" at the time. We have chosen Likelihood fields for range finder for the measurement model in this work.

Likelihood fields for range finders is a common measurement model in mobile robotics. It lacks a conceivable physical explanation. In fact, it is an "ad hoc" algorithm that does not necessarily compute a conditional probability relative to any meaningful generative model of the physics of sensors. However, the approach works well in practice. We implemented this model with slight change in the main structure. Section 4.3 explains this modified algorithm.

## Step 4: Re-sampling

The last step of this approach is resampling. In this step particles are resampled according to the particle weights. We are using roulette-wheel selection for this purpose.

### All Together

Algorithm 3 shows the basic MCL algorithm, which is achieved by substituting the proper probabilistic motion and measurements models into the algorithm particle filters. The scale estimation is obtaining along with particles resampling.

## 4.2 Motion Model

In this Section, we explain the motion model, which is implemented in the scale-invariant localization technique, in detail. The main idea of this model is similar to the motion sample odometry model. This algorithm samples from $p(x_t|m, u_{1:t-1}, z_{1:t})$ rather than a closed-form expression for computing $p(x_t|m, u_{1:t-1}, z_{1:t})$ for any $x_{t-1}$ , $u_t$, and $x_t$.

Normally, the sample motion model odometry algorithm, accepts an initial pose and an odometry reading as input, and outputs a random distribution according probability of new pose respect to odometry and previous pose. However, in this problem, odometric commands are in the metric unit and the map is made out of pixels (grid cell). Therefore, the algorithm needs to convert odometry commands from metric unit to the pixel unit based on each particle's scale factor. The aim is to personalize control command for each particle. Figure 4.1 shows one example of personalization of robot control command.

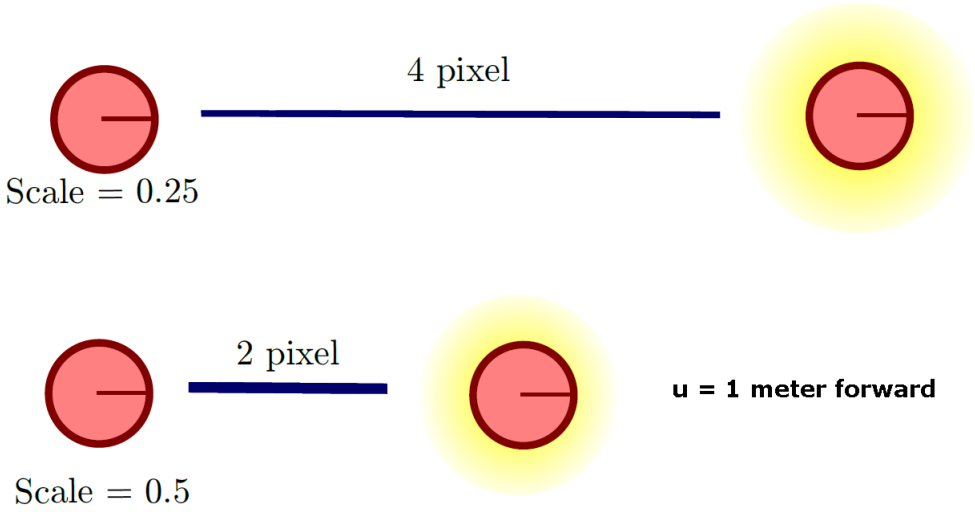



Figure 4.1: For example, a motion command is demanding the robot to move $1m$ forward. Two different particles with different scale factor behave differently with respect to the motion command. The one with scale factor of 0.25 moves 4 pixels forward and another with scale factor of 0.5 moves 2 pixels forward. The figure illustrates how the uncertainty grows as the robot moves relative to the scale factor.

The control command is formatted in such a way that the current robot's pose is comparable with the previous robot pose at any given time. The previous orientation and position are taken to be zero and afterwards the only available variables $x'$, $y'$ and $\theta'$ compared to the previous pose are $x$, $y$ and $\theta$. However, information on odometry such

as $\delta_{rot1}, \delta_{trans}$ and $\delta_{rot2}$ can be easily calculated with basic mathematical equations.

$$\delta_{rot1} = atan2(y' - y, x' - x) - \theta \tag{4.2}$$
$$\delta_{trans} = \sqrt{(x' - x)^2 + (y' - y)^2} \tag{4.3}$$
$$\delta_{rot2} = \theta' - \theta - \delta_{rot1} \tag{4.4}$$

To model the motion error, in the motion model, we assume that the real values of the rotation and translation are obtained from the measured ones by subtracting independent noise $\varepsilon_b$ with zero mean and variance $b$:

$$\hat{\delta}_{rot1} = \delta_{rot1} - \varepsilon_{\alpha_1}|\delta_{rot1}| + \alpha_2|\delta_{trans}| \tag{4.5}$$
$$\hat{\delta}_{trans} = \delta_{trans} - \varepsilon_{\alpha_3}|\delta_{trans}| + \alpha_4|\delta_{rot1} + \delta_{rot2}| \tag{4.6}$$
$$\hat{\delta}_{rot2} = \delta_{rot2} - \varepsilon_{\alpha_1}|\delta_{rot2}| + \alpha_2|\delta_{trans}| \tag{4.7}$$

The parameters $\alpha_1$ to $\alpha_4$ are robot-specific error parameters, which show clearly the error accrued with motion. We modeled the scale error in the same format. Assume that the real value of the scale factor is obtained from the previous scale factor by subtracting independent noise $\varepsilon_b$ with zero mean and variance $b$. It is necessary to define a new robot-specific error parameter, which is accrued in scale estimation. This parameter is denoted by $\alpha_5$.

$$\hat{scale} = scale - \varepsilon_{\alpha_5} \cdot scale \tag{4.8}$$

Consequently, the true position, $x_t$ , is picked up from $x_t - 1$ by an initial rotation with angle $\delta_{rot1}$, followed by a translation with distance $\delta_{trans}$, followed by another rotation with angle $\delta_{rot2}$. So,

$$\begin{pmatrix} x' \\ y' \\ \theta' \\ scale' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \\ scale \end{pmatrix} + \begin{pmatrix} \hat{\delta}_{trans}\cos(\theta + \hat{\delta}_{rot1})/scale \\ \hat{\delta}_{trans}\sin(\theta + \hat{\delta}_{rot1})/scale \\ \hat{\delta}_{rot1} + \hat{\delta}_{rot2} \\ \hat{scale} - scale \end{pmatrix} \tag{4.9}$$

Algorithm 1 shows the modified sample motion algorithm in the presented work.

## 4.3 Measurement Model

**Likelihood Fields for range finders**

The main idea is to first project the end points of a sensor scan $z_t$ into the global coordinate space of the map. The knowledge of the robot's coordinate system in the world is needed, due to projection of laser end points $z_t$ in the global coordinate frame of the map $m$. As it mentioned before, the robot's pose is depicted by $\boldsymbol{x_t} = (x, y, \theta, scale)$. $x$ and $y$ are in pixel unit, which gives us the knowledge of the robot's coordinate system in the grid-based map. In this work, we assume that the sensor's location and the robot's location are same. The angular orientation of the sensor beam relative to the robot's heading is depicted by $\theta_{k,sens}$. The measurement end-points $z_t$ can be mapped into the global coordinate system via the obvious trigonometric transformation. However, sensor data are provided in metric unit, so we can divide the beam's length by the scale factor

**Algorithm 1** modified sample motion model odometry

**procedure** SAMPLE MOTION MODEL$(u_t, x_{t-1})$
  $\delta_{rot1} = atan2(y' - y, x' - x) - \theta$
  $\delta_{trans} = \sqrt{(x' - x)^2 + (y' - y)^2}$
  $\delta_{rot2} = \theta' - \theta - \delta_{rot1}$

  $\hat{\delta}_{rot1} = \delta_{rot1} - \boldsymbol{sample}(\alpha_1\delta^2_{rot1} + \alpha_2\delta^2_{trans})$
  $\hat{\delta}_{trans} = \delta_{trans} - \boldsymbol{sample}(\alpha_3\delta^2_{trans} + \alpha_4\delta^2_{rot1} + \alpha_4\delta^2_{rot2})$
  $\hat{\delta}_{rot2} = \delta_{rot2} - \boldsymbol{sample}(\alpha_1\delta^2_{rot2} + \alpha_2\delta^2_{trans})$
  $\hat{scale} = scale - \boldsymbol{sample}(\alpha_5 \cdot scale)$

  $x' = x + \hat{\delta}_{trans}\cos(\theta + \hat{\delta}_{rot1})/scale$
  $y' = y + \hat{\delta}_{trans}\sin(\theta + \hat{\delta}_{rot1})/scale$
  $\theta' = \theta + \hat{\delta}_{rot1} + \hat{\delta}_{rot2}$
  $scale' = \hat{scale}$

  **return** $x_t = (x', y', \theta', scale')^T$

(see equations 4.10,4.11).

$$x_{z_t^k} = x + z_t^k \cos(\theta + \theta_{k,sens})/scale \tag{4.10}$$
$$y_{z_t^k} = y + z_t^k \sin(\theta + \theta_{k,sens})/scale \tag{4.11}$$

**Measurement noise.** Noise appearing from the measurement process is created by employing Gaussian distribution. In $2D-space$, this is done by finding the nearest barrier in the map. Parameter $\boldsymbol{dist_{pixel}}$ expresses the Euclidean distance in pixel between the adjacent object in the map $\boldsymbol{m}$ and the measurement coordinates $(x_{z_t^k}, y_{z_t^k})^T$. We need to convert the distance to Euclidean distance by multiplying to the scale factor.(see figure 4.2)

$$dist_{meter} = dist_{pixel} \cdot scale$$

The probability of sensor measurement is provided by a *zero-centered* Gaussian displaying the sensor noise:

$$P_{hit}(z_t^k | x_t, m) = \epsilon_{\sigma_{hit}}(dist_{meter}) \tag{4.12}$$

**Failures.** Failures happen with laser range finders when sensing black, light-absorbing objects, or measurement in bright light. A common outcome of sensor failures is max-range measurement: the sensor returns its maximum permissible value $z_{max}$. This is modeled by a point-mass distribution $P_{max}$

$$P_{max}(z_t^k | x_t, m) = I(z = z_{max}) = \begin{cases} 1 & if\ z = z_{max} \\ 0 & otherwise \end{cases} \tag{4.13}$$

**Unexplained random measurements.** Range finders rarely produce vague measurements. For example, sonars often produce illusion readings when they fly back off walls, or when they are liable to cross-talk between separate sensors. To keep everything uncomplicated, such measurements will be created using a uniform distribution spread over the entire sensor measurement range $[0; z_{max}]$:

$$P_{rand}(z_t^k|x_t, m) = I(z = z_{max}) = \begin{cases} \frac{1}{z_{max}} & if\ 0 \leq z_t^k < z_{max} \\ 0 & otherwise \end{cases} \tag{4.14}$$

Algorithm 2 illustrates the modified likelihood field range finder model. The function "***prob***" computes the probability of $\boldsymbol{dist^2_{meter}}$ under a zero-centered Gaussian distribution with variance $\boldsymbol{\sigma^2_{hit}}$ for each laser beam. We achieve the weight of each particle, by multiplying all laser beam probabilities.

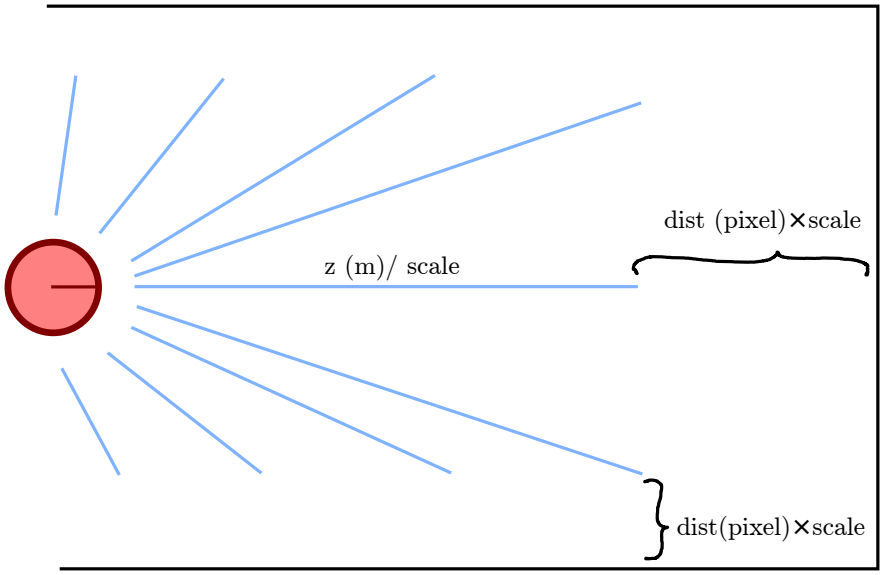



Figure 4.2: Each particle projects end points of laser beams into the map after conversion from metric unit to the pixel. The Euclidean distance between the closest barrier and laser end point must be converted from pixel to the metric unit.

**Algorithm 2** modified likelihood field range finder model

**procedure** MEASUREMENT MODEL$(z_t, x_t, m)$
  $q = 1$
  **for** all $k$ **do**
    **if** $z_t^k \neq z_{max}$ **then**
      $x_{z_t^k} = x + z_t^k \cos(\theta + \theta_{k,sens})/scale$
      $y_{z_t^k} = y + z_t^k \sin(\theta + \theta_{k,sens})/scale$
      $dist^2_{pixel} = \min_{x',y'}\{(x_{z_t^k} - x')^2 + (y_{z_t^k} - y')^2 | \langle x', y' \rangle\ occupied\ in\ m\}$
      $dist^2_{meter} = dist^2_{pixel} \cdot scale^2$
      $q = q \cdot (z_{hit} \cdot \mathbf{prob}(dist^2_{meter}, \sigma^2_{hit}) + \frac{z_{random}}{z_{max}}$
  **return** $q$

**Algorithm 3** basic Monte Carlo Localization based on particle

1: **procedure** MCL($\chi_{t-1}, z_t, u_t, m$)
2: $\bar{\chi}_t = \chi_t = \varnothing$
3: **for** $m = 1 \to M$ **do**
4: $x_t^{[m]} =$ ***modified sample motion model odometry***$(u_t, x_{t-1}^{[m]})$
5: $w_t^{[m]} =$ ***modified likelihood field range finder model***$(z_t, x_t^{[m]}, m)$
6: $\bar{\chi}_t = \bar{\chi}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
7: **for** $i = 1 \to M$ **do**
8: *draw* $i$ with probability $\propto w_t^{[i]}$
9: add $x_t^{[i]}$ to $\chi_t$
10: **return** $\chi_t$

# Chapter 5

# Map Update in Sketch-based Maps

In this Chapter, we discuss another problem in localization of a mobile robot in weakly-defined prior maps. We are focusing on a sketch-based map, which is provided by a user from the environment. The scale of the map is unknown. This map lacks accuracy, and it does not hold exact metric information of the building since data have been hand drawn by a human. The goal is to estimate the scale of the map and correct the position of the wall and objects based on the laser measurements, knowing the exact position of the robot. The robot operates to synchronize map information with measurements obtained from Laser Range Finder. The result leads to improve map specifications, which would include room dimensions, and the location of large objects relative to the walls.

## 5.1 Approach

In this problem, a sketch-based map of the environment is available which is obtained by a user about a building. At this stage, we assume the given map has been converted to the polygonal format. Therefore, the map is represented in a series of points that are connected. Our approach to solve this problem has four steps.

**Step 1:** Representing the map in homogeneous coordinates

**Step 2:** Initial guess for the entire scale of the map

**Step 3:** Data association between laser data and walls in the prior map

**Step 4:** Update the wall based on Laser points: **A)** The Kalman filter method **B)** Geometrical Constraints

The first step is to represent the polygonal map in the homogeneous coordinates. Homogeneous coordinates are very useful in computer graphics because they allow us to implement common operations such as translation, scaling, rotation by a single matrix multiplication. They also provide a scale invariant representation of points in the Euclidean plane. This is the main reason for us, to use them for map representation.

The next step is an initial guess for the scale of the entire map. The initial guess of the map helps to scale the map as close as possible to the real size of the world. This action happens only in the first step and leads to a better association between laser data and walls in the prior map.

The third step is to find the association between laser point data and walls in the prior map, and the last step is to update the location and orientation of walls based on laser data. We are proposing two methods for this problem. One is based on Kalman filter, and the other is based on Geometrical Constraints. In the following sections, all these steps have been explained in detail.

## 5.2 Map Representation

The map sketched by a user is composed of lines that give nothing more than a basic representation of the walls and large objects in the building. For instance, a box would represent a table in the middle of the room, and doors are represented by a single line segment.

In this section, we assume that the prior map is made out of a series of successive points that forms a polygon when connected consecutively. This polygon is similar to the sketch map which has drawn by the user. We use multiple polygons to demonstrate the interior details of the building and large objects. Ultimately, one polygon gives a representation of the exterior walls of the building. Figure 5.1 gives a general overview of the prior map in this experiment.


Figure 5.1: Polygonal map in homogeneous coordinates, converted from a sketch-map provided by user

Although, the prior map is a polygon, all points in the polygon have been transformed from the Euclidean coordinate to Homogeneous coordinate system. Afterwards, we replace polygons with a series of consecutive lines which connect the points of polygons. A simple form of this replacement is illustrated in figure 5.2. Representing the map in the form of successive lines in the Homogeneous coordinate system is highly favored over successive points in the Euclidean coordinate system. The advantages of Homogeneous coordinate system is discussed in Section.3.3. The use of successive lines offers a number of advantages over successive points:

1. Ray casting inside a map made out of lines is easier than a map made out of points. The point of incidence in ray casting simply obtains from the intersection of lines by defining laser beams as a line.

2. Updating map via Kalman filter, or constraints, is easier in the line format rather than point format. Assume each line stands for a wall, and then map update is a matter of relocation of lines. By moving a line forward or backward or a change in its orientation, the room's corners will be updated automatically by finding the intersection of lines in each step. In contrast, we do not have such a freedom while

representing the map in the point format, as changing the coordinate of points cause a change in two lines simultaneously.

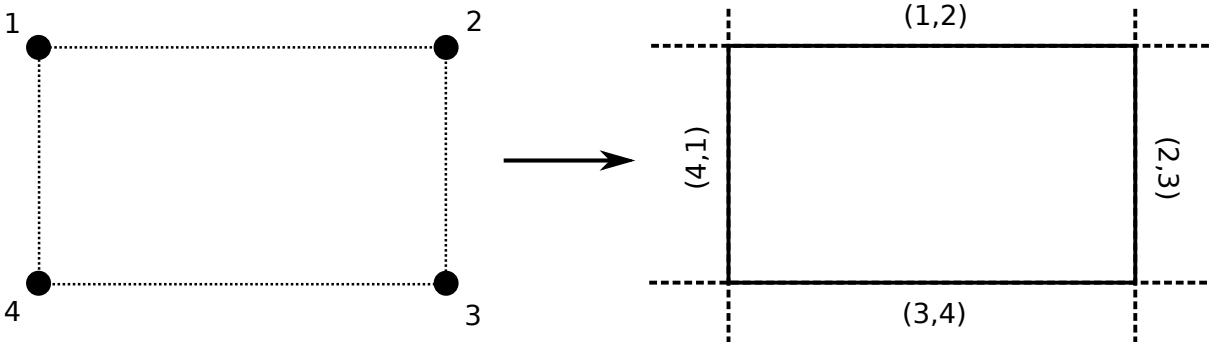



Figure 5.2: As an example, a room is depicted with 4 points. We can maintain the map in line format instead of polygon by replacing the points with lines, which connect two consecutive points. This will help us to push and pull the walls and relocate them without concerning the corners.

We added zero centered Gaussian noise to the points in the polygon to simulate the uncertainty in the prior knowledge of the environment. Consequently, all line segments in the map, carry on a covariance matrix. The Equation 3.27 calculates the covariance of each line segment based on two noisy points which the line made of.

## 5.3 Initial guess for the entire scale of the map

In this step, we need to guess approximately the entire scale of the map. Because the robot's exact pose is available, one basic method that can estimate the map scale is a comparison between ray casting information and laser measurements. Therefore, the robot propagates the rays which are similar to one single scan of laser sensor, and finds the intersection of rays to the nearest line segment in the map. The summation of the laser range lengths divided by the sum of ray lengths can give us an approximate scale. We multiply all elements of the map to the scale matrix (Eq 5.1).

$$
\begin{aligned}
Ray\ length &= \sum_i l_i \\
Range\ length &= \sum_i z_i \\
Scale &= \frac{Range\ length}{Ray\ length} \\
Scale\ matrix &= \begin{pmatrix} Scale & 1 & 1 \\ 1 & Scale & 1 \\ 1 & 1 & Scale \end{pmatrix}
\end{aligned}
\tag{5.1}
$$

Then the map size becomes much closer to the real size of the environment. The advantage of initial scale estimation is to find a better association between laser measurements and location of walls and objects in the prior map. Figure 5.3 illustrates the principle of this action in a simple form.

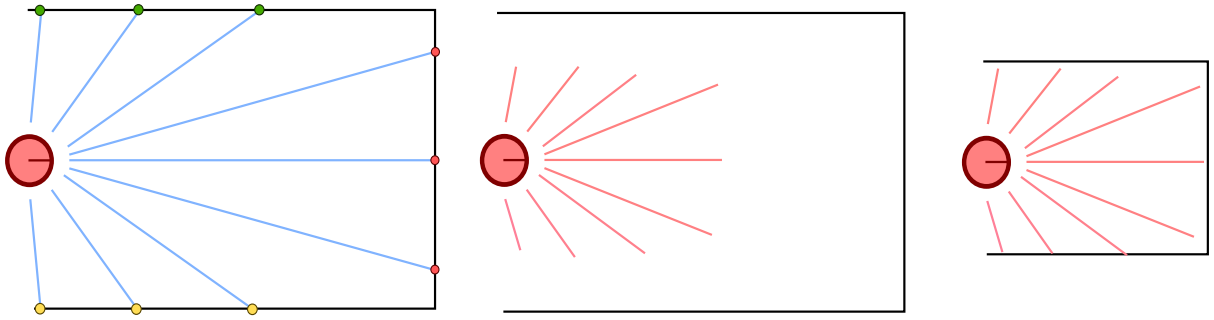


Figure 5.3: A comparison between a ray casting and laser measurements when the robot's exact pose is given, can help the robot to estimate the scale of the map. ***Left:*** Ray casting. ***Middle:*** Laser measurement. ***Right:*** Map is adjusted after Multiplication of all elements of the map to the scale matrix

## 5.4 Data Association: Between laser data and walls in the prior map

At this stage map scale is approximated and robot pose information are available to begin updating the wall's location. Robot repeats ray casting to take samples of its surrounding environment. This is similar to laser range finder in order to find the association of laser data and walls in the map. An example of this is illustrated in Figure 5.4. The points of intersection, between the rays and walls are then categorized. The points on a single wall are grouped into one set, which forms a basis for classification of laser test points. In other words, each set is composed of all points that lie on the same line. Then each laser point finds its nearest neighbor among candidates based on the Mahalanobis distance between laser points and each set.

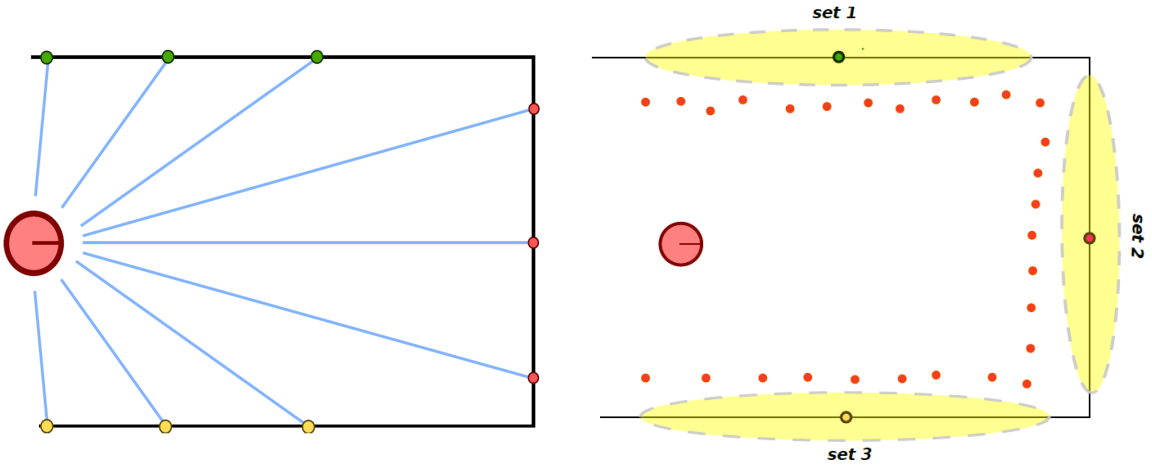



Figure 5.4: Each candidate line replaced by mean of the first point and last point of its sample and the covariance matrix of all sample points depicts the uncertainty of wall's location. Small red points demonstrate laser point data. Each point belongs to the nearest neighbor set according to its Mahalanobis distance to the sets.

The Mahalanobis distance is a descriptive statistic that provides a relative measure of a data point's distance (residual) from a regular point. It also calculates similarity of an unfamiliar sample set to a known one. It varies from Euclidean distance in that it is scale-invariant, and the correlations of the data set are taken into account. In an official manner, the Mahalanobis distance of a multivariate vector $x = (x_1, x_2, x_3, \ldots, x_N)^T$ from a set of values with mean $\mu = (\mu_1, \mu_2, \mu_3, \ldots, \mu_N)^T$ and covariance matrix $\Sigma$ is finding out as:

$$D_M(x) = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)}.$$

The sample points are laying on a line, therefore, the uncertain nature of data was not taken into account. Consequently, a zero-centered Gaussian noise is added to the covariance of each set to obtain a higher quality association. Additional noise improves the probabilistic nature of the proposed method.

Often, the nearest neighbor set to a laser point seems too far. As a result, a threshold must be defined that can exclude outliers to avoid wrong association. At the final stage of this section, we have classified all laser points and found their corresponding wall on the map.

## 5.5 Update the wall based on Laser points

All previous steps are pre-conditions to this final stage of map updating. As mentioned earlier, the prior map does not hold precise information from the environment. The rooms dimensions and angles between the walls do not offer 100% precision. What follows is a discussion of two methods to update the prior map. The first method is based on Kalman filter, and the second is based on geometrical constraints.

### 5.5.1 Map Update via Kalman Filter

The Kalman filter is an iterative linear update mechanism employed to maximize posterior probability estimation. The parameters to be estimated in the application of a linear dynamical system change with time.

Initial algorithm of the Kalman filter is composed of two steps of prediction and update. The theoretical details of the two steps were discussed in Section.3.2. The Kalman filter used in this work differs slightly from the original algorithm. At this stage, it is assumed that location of walls must be updated according to their corresponding the laser data points; therefore, eliminating the need for any prediction step. In other words, walls are taken to be static without changing direction and no control command can be applied to them. Hence, details remain the same if prediction were performed.

During the second stage of the algorithm, update, each wall that is chosen in the ray casting step is assumed as a prior line. The laser data points replaced with their regression line.

A *3-parameter* vector stands for each prior line, with a *3-by-3* covariance matrix is calculated by equation 3.25 to 3.27. Same dimension has been implemented in regression line (see eq.3.45 for covariance of the regression line).

The major challenge is Innovation (or residual) covariance matrix inversion. Both covariance matrices for regression line and prior line are a singular matrix of *rank-2*.

Therefore, the updated line does not appear between prior and regression line. Kanatani suggests implying a rank-constrained generalized inverse [24].

Using the Kanatani's method does not necessarily lead to an acceptable result. For instance, updated line may not fall within the range set by prior line and the regression line. Finally, a better matrix inversion method can be achieved by employing the Moore-Penrose pseudo inverse with a tolerance for eigenvalues. If an eigenvalue is smaller than tolerance, it will be eliminated from the pseudo inverse calculation. The tolerance value is achieved by experience. Figure 5.5 presents a simplified model for the Kalman filter.

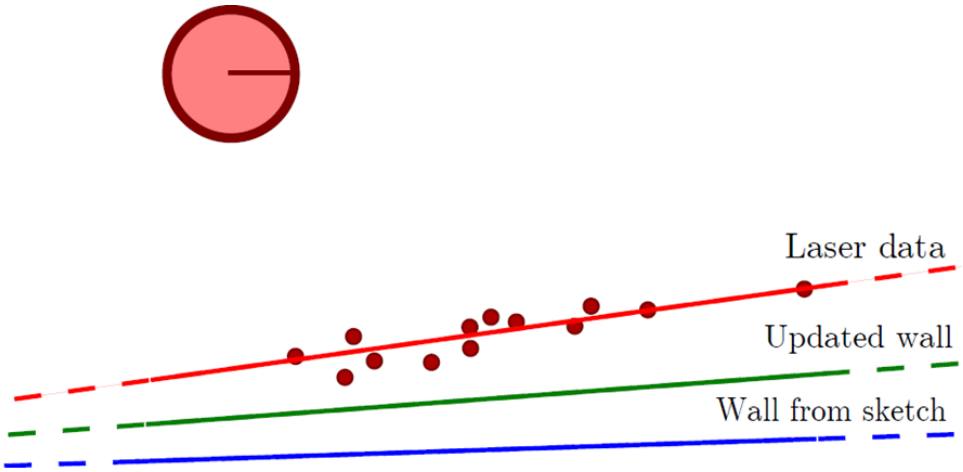



Figure 5.5: Wall update via Kalman filter: Red line is the regression of observation(laser points). The blue line is the correspondence prior (wall in the map). Green line stands for the updated wall

### Deming Regression for Kalman Filter

In statistics, fitting a line to approximately linearly correlated data is called regression; however the independent variables (often called predictor variables) and the dependent variables (also called observed variables) must be specified. In this work, the $x$ and $y$ coordinates are totally equivalent when we fit a line to approximately collinear laser data. The coordinate axes are chosen simply for computational convenience, and the line to be fitted must be identical regardless of coordinate system used [24].

Implementing of Deming regression instead of linear regression follows similar reasoning. Deming regression is a particular state of total least squares, which provides for any number of predictors and a more complex error structure. (See Section 3.4)

Classification of laser measurement leads to a common assumption that each set represents a geometric line. Using regression at this stage helps us gain a better comparison between prior line and measurement data. Calculation of fitted line's covariance is a major challenge. The uncertainty of a fitted line to data points is represented by a covariance matrix. A geometrical line has two degrees of freedom, but the covariance matrix for a line is a $rank - 2$ singular 3 by 3 matrix. Equation 3.45 is given for the calculation of this matrix.

### Detecting outliers

Outlier detection is a critical step in many data-mining applications. Mahalanobis distance is often used to detect outliers, particularly in the extension of regression models. A point that has a significant Mahalanobis distance from the rest of the group of points is said to have greater leverage. It has a bigger impact on the slope or coefficients of the regression line. Additionally, Mahalanobis distance is also used to find multivariate outliers. Regression methods can be applied to define whether a particular case within a sample group is an outlier.

There are different ideas regarding which cutoff values to use for recognizing most influential points. A straightforward operational guideline of ($D_{mahal} > 4$) has been suggested in this work. This threshold is based on experience only, while function output is more appealing.

## 5.5.2 Map Update via Geometrical Constraints

In this part, we discuss the second approach for updating walls in the polygonal map. This method is based on a geometrical constraint which has been proposed by Kanatani. Kanatani is a Japanese mathematician that represents this principle in his book "Statistical Optimization for Geometric Computation" [24]. We introduce the principle of this method in Section 3.5

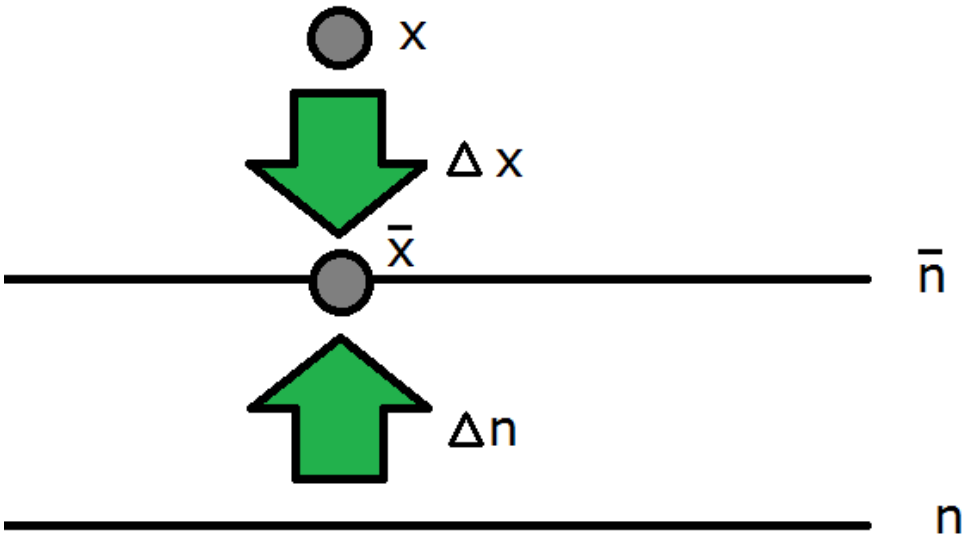



Figure 5.6: Constraint for Incidence of a point to a line

### Approach

Assume we have a point $\boldsymbol{x}$ that is an estimate of a point $\bar{\boldsymbol{x}}$. And we have line $\boldsymbol{n}$ that is an estimate of a line $\bar{\boldsymbol{n}}$. Suppose $\bar{\boldsymbol{x}}$ lies on $\bar{\boldsymbol{n}}$ in the absence of noise, then a problem arises for optimally correcting them so as to make them incident. The question is how much we have to move the line $\boldsymbol{n}$ to optimally correct it and put it on the $\bar{\boldsymbol{n}}$.(See figure 5.6)

Same idea happens to the point $\boldsymbol{x}$ , how much is the $\boldsymbol{\Delta x}$ that moves point $\boldsymbol{x}$ to the point $\bar{\boldsymbol{x}}$. If $\boldsymbol{x}$ has a covariance matrix of $\boldsymbol{V}[\boldsymbol{x}]$ and $\boldsymbol{n}$ has a covariance matrix of $\boldsymbol{V}[\boldsymbol{n}]$

then this is an optimization problem minimizing $\boldsymbol{J}$(equation 3.49) under linear constraint in Equation 3.50.

Assume a laser point is corresponding to a wall in the prior map. The idea is if the laser point corresponds to a wall, they should be incident to each other in the absence of noise. The aim is to find out how optimally correcting them so as to make them incident. This can be regarded as a constraint for map updating. The principle is presented in Section 3.5.

Instead of using regression of laser points, we are dealing with all laser points in a set which is corresponding to the candidate wall for an update. Therefore, a single wall will be updated several times by each laser point in the set.

Fortunately, in this process, matrix inversion is not taking place. Hence, there is no need to handle Moore-Penrose pseudo inverse. The number of updates of any wall is much more than the Kalman filter method, however the running time per step is almost half.

Figure 5.7 shows a simplified example of the constraints update. Red points represent laser points with added Gaussian noise. The blue line corresponds to the prior wall in the map with the associated uncertainty represented by light blue lines. The green line is the final state of updated wall composed of all updates.

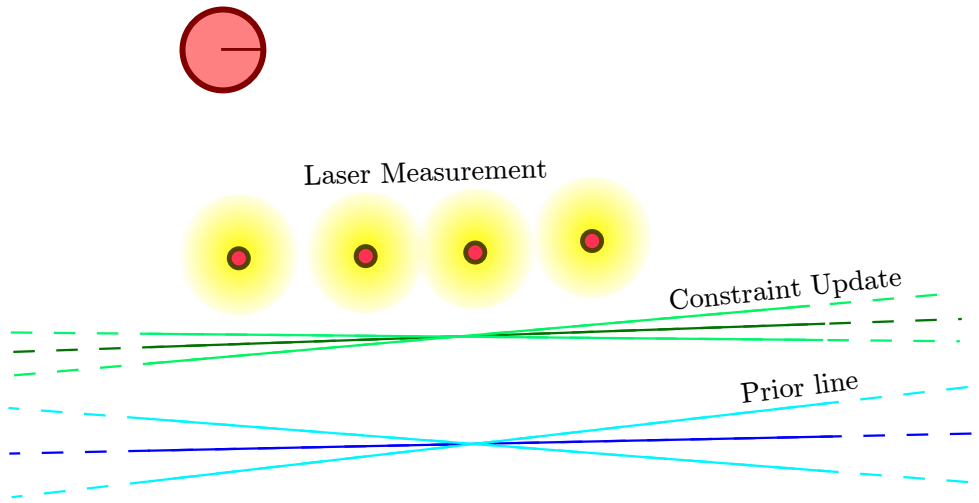



Figure 5.7: Wall update via geometrical constraint: red points stand for laser points with an arbitrary initial Gaussian noise. The blue line is the prior line with a relatively high uncertainty. Green line stands for updated wall with a lower uncertainty relative to prior line.

It is important to note that there are as many updates corresponding to the number of laser points in this model. For instance, in the figure 5.7, the prior line has been updated four times to give the green line. This type of constraint can only deal with a single point and a single line at a time. Therefore, in the algorithm used a loop has been employed to consider all laser points that correspond to the prior wall.

All the techniques used in the optimization of the Kalman update are used here again. The Mahalanobis distance is one example that can be employed to identify outlier. Using this technique makes updated wall very close to their real position in the environment.

Furthermore, a threshold for updating the location of wall has been defined. Whenever the angle between the constraint update wall and prior wall is greater than 15 degrees,

the algorithm omits this update. The algorithm considers this value to be a calculation error in data association.

# Chapter 6

# Experiments and Results

## 6.1 Localization in a Map with Unknown Scale

After several experiments with different number of particles, we discovered that increasing the quantity of particles does not guarantee more accurate scale estimation. As an example, within the best case scenario, 1500 particles predict the scale of map with least standard error and closest to the real size of the environment, however, in the same experiment with 3000 particles the results weakened. Figure 6.1 shows the results by implementing different number of particles.

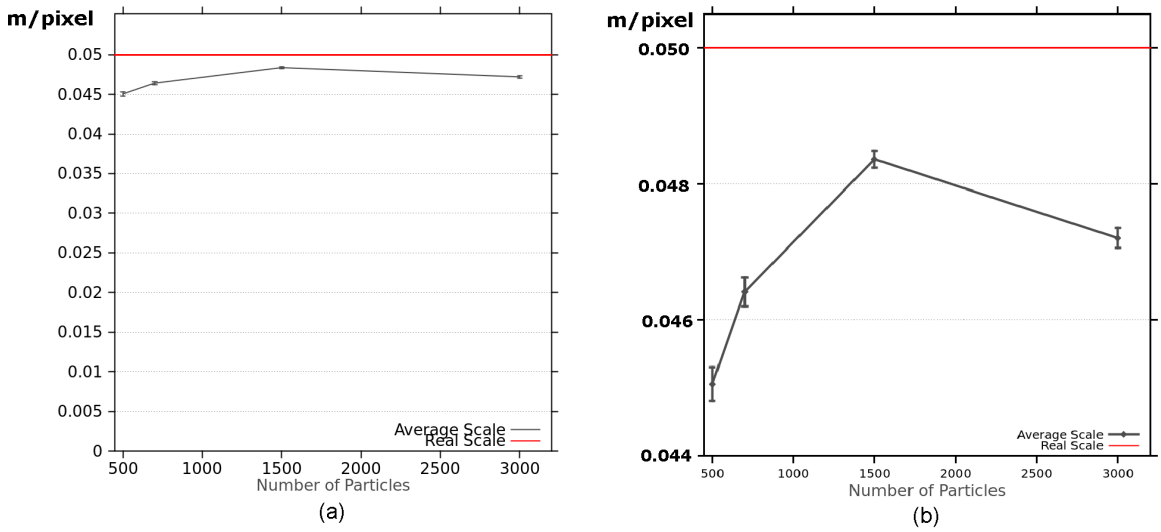



Figure 6.1: Scale estimation result: a) the real scale of map is depicted by the red line. The scale estimation is very close to real scale therefore the plot is not very understandable. b) The information on (a) is shown in a smaller range for better understanding of standard error.

The number of time steps to particles convergence to the robot's location varies in different experiments. The table 6.1 illustrates the approximate number of time steps for localization and scale estimation respect to the quantity of particles and approximated run time for every time step.

Not all the experiments succeed to estimate properly the scale of the map. In Some

Table 6.1: Average number of time steps to localize the robot

| Number of particles | 500 | 700 | 1500 | 3000 |
|---|---|---|---|---|
| #-Step to Converge | 113 | 106 | 97 | 111 |
| Each time-step(sec) | 0.38 | 0.53 | 1.63 | 9.6 |

tests with this technique, the particles converge in a symmetric position with respect to the real position of the robot; though, scale estimation is nearly close to the real one. Figure 6.2 presents the percentage of success in several experiments for various numbers of particles.

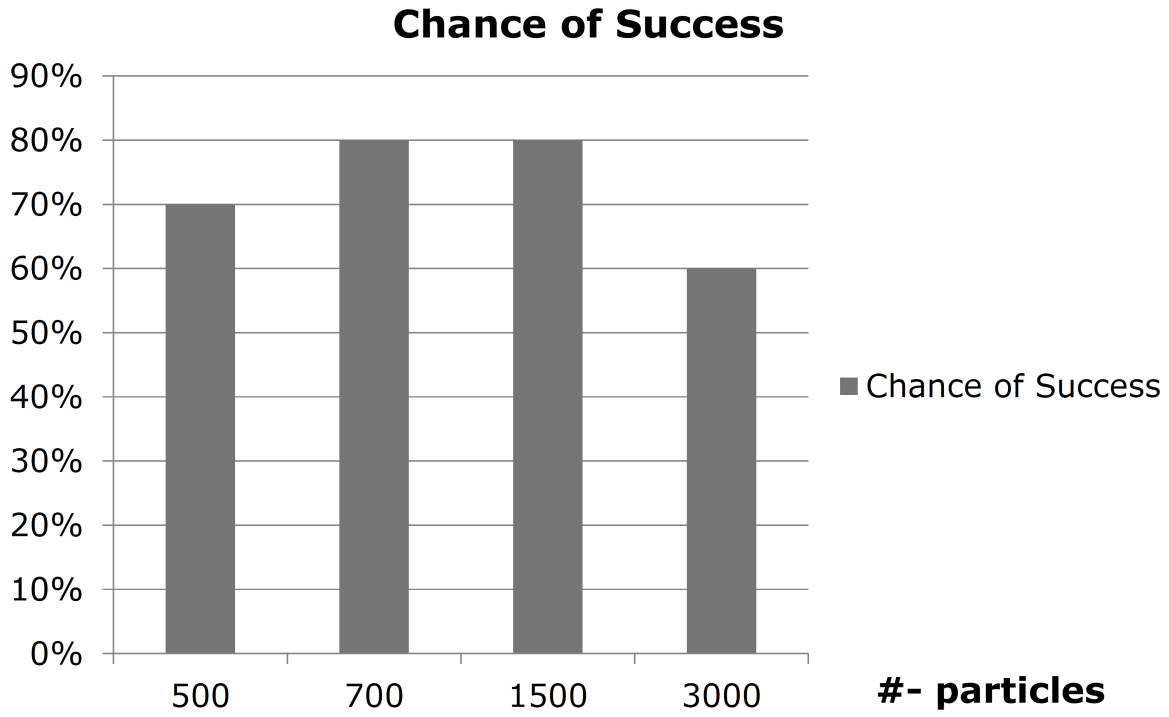
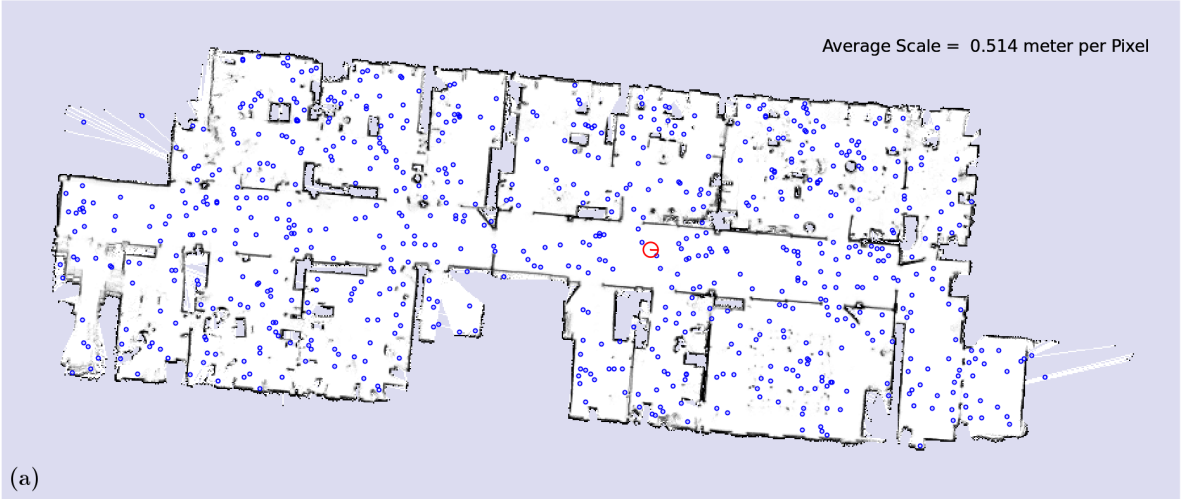



Figure 6.2: This figure shows the chance of success for scale estimation and localization of the robot in the expecting time step. In some experiment, the robot is localized finally but long after the expected time step.

The method implemented in this section has proven almost successful in robot localization and the estimation of the map scale after about 100 time steps. The average scale factor represented by each particle is remarkably close to the real scale of the map that is 0.05 $m/pixel$. Occasionally, particles lose their convergence for a short time. This happens while there are several small objects around the robot, especially in rooms and when the robot is turning. The correct location, however, is found after a few steps and the scale estimation is close enough to the real value.

Figures 6.3 show a few steps between step 0 and step 100 for a test for 700 particles. The particles with higher scale factor may jump from a room to another room, simply because the control command has multiplied with their scale factor. However, after localization, the movement of particles is looking more natural.


Average Scale = 0.514 meter per Pixel

(a)


Average Scale = 0.035 meter per Pixel

(b)


Average Scale = 0.041 meter per Pixel

(c)


Figure 6.3: Robot in motion : a) time-step 1. b) time-step 50. c) time-step 80. d) time-step 98.

## 6.2 Updating the sketch map

In this section we discuss the experiment and the result of polygonal map updating with two different methods: 1) Kalman filter method. 2) Geometrical constraints. At the end, we compare these two methods based on the updated map after around 1400 time steps. Knowing the exact location of the robot helps categorize laser data and find their correspondence in the prior map. Considering all that has been discussed thus far, correcting the polygonal map can take place according to the following methods.

### 6.2.1 The Kalman filter method

In Figure 6.4, one successful update of the Kalman filter method is illustrated. The zoomed in region shows a part of the wall, drawn by the user, at the eastern side of the building which is drawn in the wrong orientation. The pink points represent laser data whereas laser points corresponding to the wall are shown in black. The regression line of laser data is represented by the red line, while the blue line denotes prior wall. The green line is the updated wall by Kalman filter function.

Another example of a Kalman filter update is given in Figure 6.5. Laser data plotted on the map are shown in pink - part $(a)$. The location of the wall on the polygonal map is slightly behind that it should be, based on laser data. In part $(b)$, the robot has identified a number of laser points that correspond with the wall. These points are shown in black. The green line expresses this functionality of the Kalman filter update. In part $(c)$, the blue line is replaced with the green line and corners are updated automatically. The nature of Kalman filter dictates the positioning of the updated wall in between the prior wall and the regression line.

In the following Figure 6.6, the last example of the functionality of the Kalman filter is illustrated. In part $(a)$, a view of the central door of the corridor in the building is given which has been drawn incorrectly on the sketch map. The frame of the door is not aligned. Pink points that show laser points indicate the correct position of this door to a good approximation. In part $(b)$, the robot updates the first line according to the

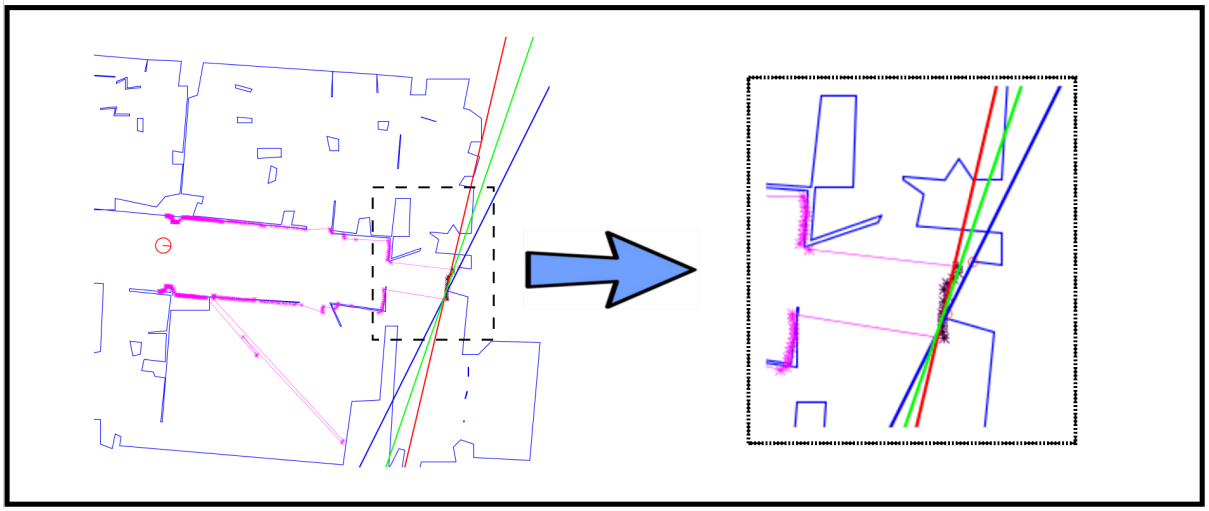


Figure 6.4: Kalman filter update (example 1): Red line stands for regression. Blue line for Prior Wall. Green line for updated wall. Pink polyline represents laser point data. Black stars are corresponding to the prior wall.

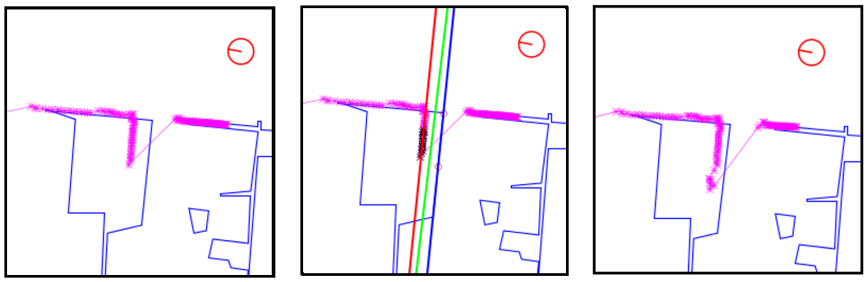



Figure 6.5: Kalman filter update (example 2): a) Laser data points are plotted in pink. The blue lines are sketched prior map. b) The black points are corresponding to the prior line. The Kalman filter updates the blue line by using the regression line (in red) as a measurement. c) The wall has moved to a new position.

method explained in the previous Figure. However, due to the parallelism of doors and interior walls with the two lines, this update only moves one of the lines and update of the next line requires more steps in part $(c)$. Therefore, part $(d)$, following similar steps from part $(c)$, updates the second line, and the final outcome is shown in part $(f)$.

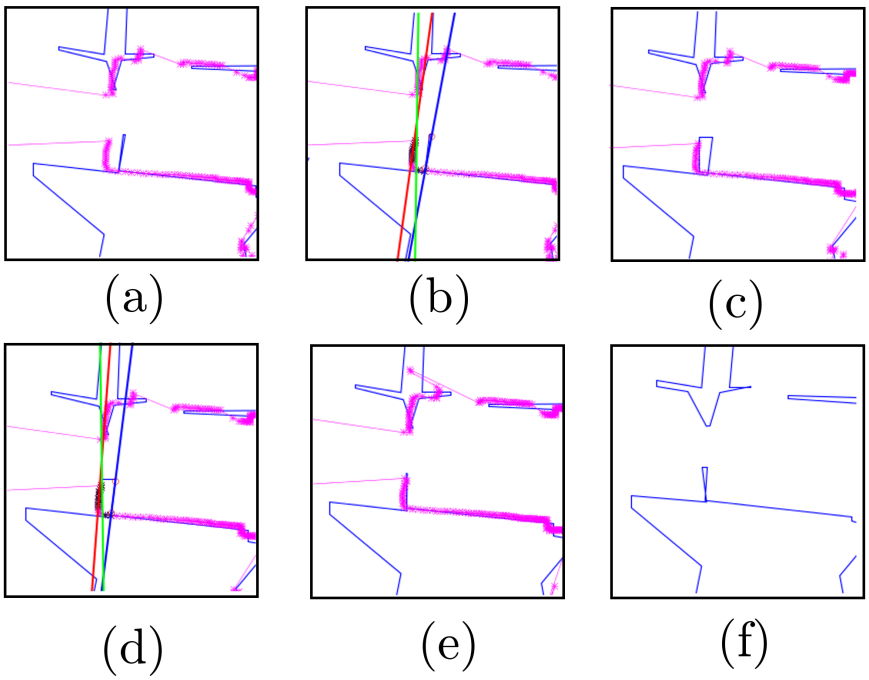



Figure 6.6: Kalman filter update (example 3): a) central door of the corridor in the building. b) Kalman filter updates the front blue line. c) Front line is updated, but the back line is still incorrect. d) The Kalman filter updates the second line. e) The central door has been corrected.

In Figure 6.7 multiple stages of the Kalman filter update are shown. The robot explored the building in 1460 time steps. The prior polygonal map is presented in part (a). Over $340^{th}$ time-step, the updated map is illustrated in part (b). In part (c), the Kalman filter updated the map for $760^{th}$ time. And finally, changes in the map can be compared with the occupancy grid map in Figure 6.8 after around 1400 time steps. There are some major errors in the final polygonal map that has been shown by red arrows. The scale of the map in general has been estimated correctly, but the major mistakes reduced the quality of final result.

It is essential to note how the robot achieves successful minor modifications during the initial stage followed by mistakes arising from the wrong data association between laser point and the prior map. Alternatively, this could be due to errors associated with the Kalman filter update.

The algorithm cannot rectify these errors by itself, and this is a significant challenge of this whole system. Various other methods for addressing this problem have been proposed in the section on recommendations and future work.


Figure 6.7: The Kalman filter update in progress: a) $1^{st}$ time-step. b) $340^{th}$ time-step. c) $760^{th}$ time-step.

Figure 6.8: The Kalman filters method's result: After 1400 time steps. It is comparable with the map provided by occupancy grid mapping. The major errors are noted by the red arrows.

### 6.2.2 Geometrical Constraints method

This method is based on more simple calculations that result in a faster algorithm run compared to the Kalman filter method. An example of the functionality of this method is given in the Figure 6.9 . In part $(a)$, a similar example to Figure 6.5 can be seen again in the same scenario. The main difference, however, is the absence of a regression line. The algorithm updates each line with the corresponding laser points. In part $(b)$, the black points are equivalent of blue prior lines and the green line is the line of constraints update. Parts $(c)$ to $(f)$ show a gradual moving forward of the robot and updating of laser data accordingly. This method performs a very steady modification in the map and accounts for minor changes in the prior map with higher accuracy in comparison to Kalman filter approach.

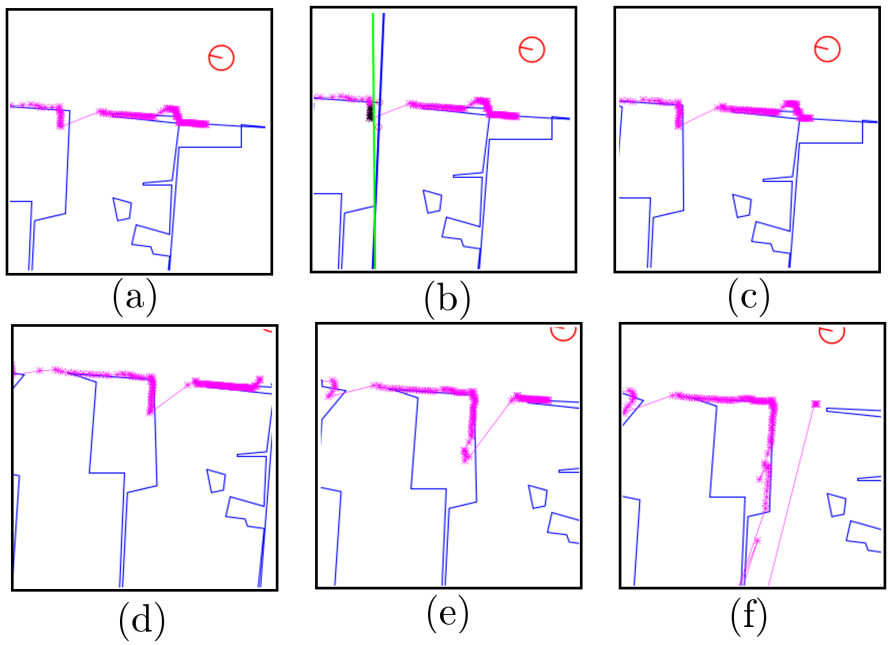



Figure 6.9: Geometrical Constraints update (example 1): a) Laser Data points are plotted in pink. Blue lines are the sketched prior map. b) The black points are corresponding to prior line. The wall is updated based on constraint. The result is the green line c) after the first update f) after fifth and last update.

In Figure 6.10, there is another example of updating by constraint where the position of the walls of the central door in the corridor is incorrect. The same example has mentioned before in the Kalman filter update section (Figure 6.6). Part $(a)$ shows the prior map without any update. Part $(b)$ is concerned with optimizing the location and relocating the candidate wall for update with respect to the associated laser data (black points). Since, the internal wall is represented by two lines, this operation is repeated in part $(d)$. Parts $(e)$ and $(f)$ confirm correction of the wall position to a good approximation.

In the figure 6.11 the updated map after several updates by constraint has been illustrated. And finally in Figure 6.12 the final result after about 1400 time steps is depicted which it can be compared with the occupancy grid map. In this stage, we have not done any quantitative comparison between the Kalman filter method's final result

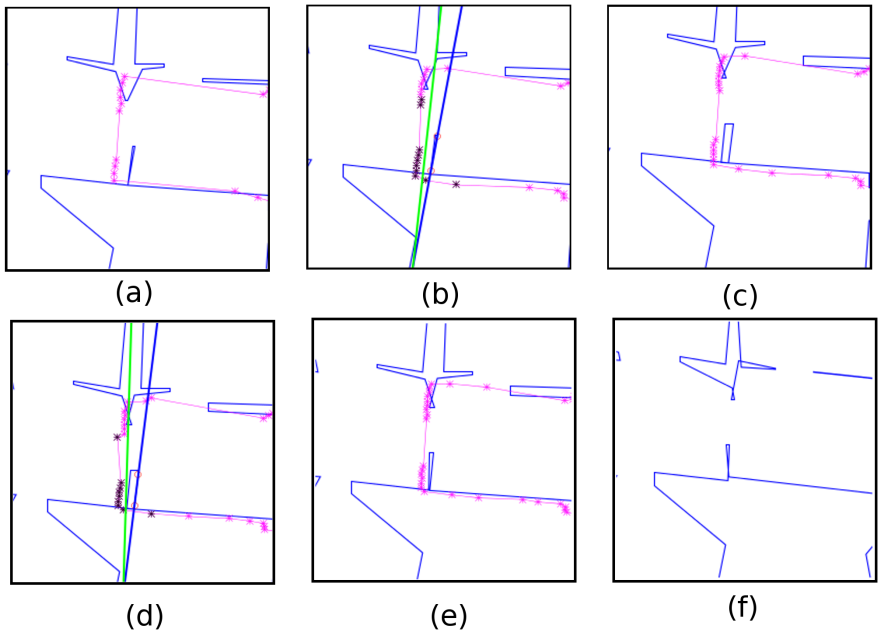



Figure 6.10: Geometrical Constraints update (example 2): a) central door of the corridor in the building. b) Constraint updates the front blue line. c) Front line is updated, but the back line is still incorrect. d) Constraints updates the second line. e) Central door has been corrected. f) Final result.

and geometrical constraints. However, based on observation, the geometrical constraint method has the following advantages:

1. In geometric constraint, errors in updates are only due to the wrong data association between laser data and prior map. However, in the Kalman filter method, additional error can be produced by regression line calculation as well. Therefore, geometrical constraints have a more reliable model.

2. In geometrical constraints, relocation of walls, response for applying required modification to the map is much smoother than Kalman filter method. In general, less major errors appear in the final result.


Figure 6.11: Constraints update in progress: a) The prior map at $1^{st}$ step. b) The map after $380^{th}$ time step. b) The map after $913^{th}$ time step.

Figure 6.12: Geometrical constraint method's result: After 1400 time steps. It is comparable with the map provided by occupancy grid mapping. The major errors are noted by the red arrows.

# Chapter 7

# Summary

## 7.1 Conclusion

This work analyses mobile robot localization in weakly-defined prior maps. There are two approaches. The first approach is a localization technique in a consistent map with unknown scale. The second approach is a map modification and scale estimation technique in a sketch based map which is prepared by a user, with unknown scale and incorrect data while the robot's exact pose is available.

The proposed method, in the first approach is based on Monte Carlo localization, which is modified for map with unknown scale. We were able to localize the robot in the given map, by augmenting the state space, while the scale of the map has been estimated approximately. Since, the final goal is to localize the robot in a sketch based map, in future work we will employ the same technique for a polygonal map.

In the second approach, robot's exact location is available in a sketch based map where the robot is responsible for correction of user's mistakes. The scale of map is not available either. A new method for polygonal based map has been represented in this work. The map represents a series of lines instead of points, which is allowing the robot to change the location of walls and objects simply by pushing and pulling lines without losing the overall map sketch.

The map is produced by a series of lines, and each line carries on a covariance matrix which enables us to represent the map with uncertainty. All segments of the map are expressed in homogeneous coordinates that coveys many advantages for map updating.

We developed a data association technique for laser points, which finds the corresponding walls, candidate for update, in the prior map based on nearest neighbor. Nearest neighbor finds the association of data based on the Mahalanobis distance. Some laser points are not able to find the corresponding wall in the prior map or associate with a wall that they do not belong to. This can be due to the presence of noise and complexity of real world with respect to simple prior sketch map. However, this technique has been improved by extracting outliers and defining a threshold for the distance between the walls and laser points. In the recommendation section, we proposed one more idea to improve the current method.

Finally, two methods have been proposed for updating the polygonal map. The first method is based on Kalman filter which estimates the location of wall by comparing the regression line of laser points and the prior wall in the map. The Kalman filter does not predict the location of the wall but only update it. The covariance matrix of updated wall is normally contains less uncertainty respect to the prior wall.

The second method is based on geometrical constraint which has a better output compared to the Kalman filter method. In this technique, instead of calculating of regression line as measurements, the prior wall updates directly by the laser data point.

Major difficulties of these two methods can be summarized to:

1. Laser data points that are associated with a wall do not necessarily lie on a straight line. This is one of weaknesses of our data association method. In addition, the level of noise causes the regression line to differ from the desired wall. However, outliers in the regression are identified and taken out of the calculations.

2. The covariance matrix for both prior lines and regression lines are singular matrices which causes calculation errors for the algorithm in update line and the inverse matrix. Therefore, the final line differs from the desired line. Although, using a special method for inverting the covariance matrix greatly reduces the scope of this problem.

3. In the geometric constraints method where instead of regression, laser data points are used directly for updating prior lines, which eliminates calculation errors associated with regression line and matrix inversion. In addition, simpler calculations are performed which result in a shorter run-time compared to the Kalman filter methods. Considering the benefits of this method in comparison with the Kalman filter method, it will be employed in our future works.

## 7.2 Recommendation

- One major improvement for categorization of laser point data and identification of the corresponding data in sketched map can be achieved through the transformation of laser points into a polygon. The method is followed by simplification of the polygon. Latecki et al. has already presented a similar idea in mathematical literature given in article [30]. Shape comparison can then, similar to [31], improve this classification/categorization.

- Occasionally, the intersection of two consecutive lines lies in infinity, due to changes in orientation of two consecutive walls which make them parallel. Therefore, the map representation technique works more efficiently when dealing with less acute angles and more right angles in the prior map. Use of complex polygons in the initial sketch is also discouraged.

## 7.3 Future Work

- One important area for further research is a combination of the first approach to the second. This combination enables particles from the first step to apply the necessary corrections into the sketched map after scale estimation and localization. This requires employing the first approach in the polygonal map at the beginning. Each particle is then assigned its own map, which eliminates the need for knowing exact robot's pose in the second approach.

- We can improve the scale estimation by using multiple sensors. The more data can be gathered from the robot's surrounding environment, the more accurate update

can be achieved. The gathered data can then be used as alternative constraints for prior map correction. It ultimately results in a more realistic final map with high precision.

- Additionally, the final map can be designed in *3D* using spatial planes instead of *2D* lines. The characteristics of elements of the map will remain unchanged while the intersections of consecutive planes form spatial corners.

# Bibliography


[1] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotic*. Intelligent Robotics and Autonomous Agents Series. Mit Press, 2005.

[2] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Active markov localization for mobile robots. *Robotics and Autonomous Systems*, 25(3):195–207, 1998.

[3] Wolfram Burgard, Andreas Derr, Dieter Fox, and Armin B Cremers. Integrating global position estimation and position tracking for mobile robots: the dynamic markov localization approach. In *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, volume 2, pages 730–735. IEEE, 1998.

[4] Leopoldo Jetto, Sauro Longhi, and Giuseppe Venturini. Development and experimental validation of an adaptive extended kalman filter for the localization of mobile robots. *Robotics and Automation, IEEE Transactions on*, 15(2):219–229, 1999.

[5] Frank Dellaert, Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Monte carlo localization for mobile robots. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 2, pages 1322–1328. IEEE, 1999.

[6] Sebastian Thrun, Dieter Fox, Wolfram Burgard, and Frank Dellaert. Robust monte carlo localization for mobile robots. *Artificial intelligence*, 128(1):99–141, 2001.

[7] Chieh-Chih Wang, Charles Thorpe, and Sebastian Thrun. Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 1, pages 842–849. IEEE, 2003.

[8] MWM Gamini Dissanayake, Paul Newman, Steven Clark, Hugh F Durrant-Whyte, and Michael Csorba. A solution to the simultaneous localization and map building (slam) problem. *Robotics and Automation, IEEE Transactions on*, 17(3):229–241, 2001.

[9] Jana Kosecka and Xiaolong Yang. Global localization and relative pose estimation based on scale-invariant features. In *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 4 - Volume 04*, ICPR '04, pages 319–322, Washington, DC, USA, 2004. IEEE Computer Society.

[10] Rainer Kümmerle, Bastian Steder, Christian Dornhege, Alexander Kleiner, Giorgio Grisetti, and Wolfram Burgard. Large scale graph-based slam using aerial images as prior information. *Autonomous Robots*, 30(1):25–39, 2011.

[11] K. Oikawa K. Terabayashi, T. Emaru and T. Tsuchiya. Navigation of autonomous robot with handwritten map. In *In Proc. of The Eight Robotics Symposia, pp.361– 366*, 2003.

[12] Marjorie Skubic, Derek Anderson, Samuel Blisard, Dennis Perzanowski, and Alan Schultz. Using a hand-drawn sketch to control a team of robots. *Autonomous Robots*, 22(4):399–410, 2007.

[13] Vachirasuk Setalaphruk, Atsushi Ueno, Izuru Kume, Yasuyuki Kono, and Masatsugu Kidode. Robot navigation in corridor environments using a sketch floor map. In *Computational Intelligence in Robotics and Automation, 2003. Proceedings. 2003 IEEE International Symposium on*, volume 2, pages 552–557. IEEE, 2003.

[14] Gaurav Parekh, Marjorie Skubic, Ozy Sjahputera, and James M Keller. Scene matching between a map and a hand drawn sketch using spatial relations. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 4007–4012. IEEE, 2007.

[15] Keisuke Matsuo and Jun Miura. Outdoor visual localization with a hand-drawn line drawing map using fastslam with pso-based mapping. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 202–207. IEEE, 2012.

[16] Michael Veeck and W Burgard. Learning polyline maps from range scan data acquired with mobile robots. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 2, pages 1065–1070. IEEE, 2004.

[17] Diedrich Wolter and Longin J Latecki. Shape matching for robot mapping. In *PRICAI 2004: Trends in Artificial Intelligence*, pages 693–702. Springer, 2004.

[18] Rolf Lakaemper, Longin Jan Latecki, and Diedrich Wolter. Incremental multi-robot mapping. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3846–3851. IEEE, 2005.

[19] Mohamad Ali Movafaghpour and Ellips Masehian. Poly line map extraction in sensor-based mobile robot navigation using a consecutive clustering algorithm. *Robotics and Autonomous Systems*, 60(8):1078–1092, 2012.

[20] Gian Diego Tipaldi. *Looking Inside for Mapping the Outside: Introspective Simultaneous Localization and Mapping*. PhD thesis, University of Rome La Sapienza, 2009. in press.

[21] O. Cappé, E. Moulines, and T. Ryden. *Inference in Hidden Markov Models*. Springer Series in Statistics Series. Springer Science+Business Media, Incorporated, 2005.

[22] Stefan Oßwald. Techniques for autonomous stair climbing with humanoid robots. Master's thesis, University of Freiburg, Department of Computer Science, Humanoid Robots Lab, February 2012.

[23] J. Bloomenthal and J. Rokne. Homogeneous coordinates. *The Visual Computer*, 11(1):15–26, 1994.

[24] Kenichi Kanatani. *Statistical optimization for geometric computation: theory and practice*. Dover Publications, Incorporated, 2005.

[25] Sabine Van Huffel and Joos Vandewalle. *The total least squares problem: computational aspects and analysis*, volume 9. Society for Industrial Mathematics, 1987.

[26] W.E. Deming. *Statistical adjustment of data*. Dover Books on Mathematics Series. Dover Publications, 1964.

[27] Wayne A Fuller. *Measurement error models*, volume 305. Wiley, 2009.

[28] Sam Roweis and Zoubin Ghahramani. A unifying review of linear gaussian models. *Neural computation*, 11(2):305–345, 1999.

[29] J.D.A. HAMILTON. *The Time Series Analysis*. Princeton Univers. Press, 1994.

[30] Longin Jan Latecki and Rolf Lakämper. Convexity rule for shape decomposition based on discrete contour evolution. *Computer Vision and Image Understanding*, 73(3):441–454, 1999.

[31] Diedrich Wolter, Longin J Latecki, and Rolf Lakämper. An architecture for shape-based robot mapping.