ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

3D SIMULTANEOUS LOCALIZATION AND MAPPING METHODS IN OUTDOOR AND LARGE-SCALE ENVIRONMENTS FOR AUTONOMOUS ROBOT NAVIGATION

Ph.D. THESIS

Cihan ULAŞ

Department of Control and Automation Engineering

Control and Automation Engineering Programme

DECEMBER 2012

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

3D SIMULTANEOUS LOCALIZATION AND MAPPING METHODS IN OUTDOOR AND LARGE-SCALE ENVIRONMENTS FOR AUTONOMOUS ROBOT NAVIGATION

Ph.D. THESIS

Cihan ULAŞ
(507072106)

Department of Control and Automation Engineering

Control and Automation Engineering Programme

Thesis Advisor: Prof. Dr. Hakan TEMELTAŞ

DECEMBER 2012

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

OTONOM ROBOT NAVİGASYONU İÇİN DIŞ VE GENİŞ-ÖLÇEKLİ ORTAMLARDA 3D EŞ ZAMANLI KONUMLAMA VE HARİTALAMA YÖNTEMLERİ

DOKTORA TEZİ

Cihan ULAŞ
(504072106)

Kontrol ve Otomasyon Mühendisliği Anabilim Dalı

Kontrol ve Otomasyon Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Hakan TEMELTAŞ

ARALIK 2012

Cihan Ulaş, **a Ph.D.** student of ITU **Graduate School of Science Engineering and Technology** student ID **504072106** successfully defended the **thesis/dissertation** entitled "**3D SIMULTANEOUS LOCALIZATON AND MAPPING METHODS IN OUTDOOR AND LARGE-SCALE ENVIRONMENTS FOR AUTONOMOUS ROBOT NAVIGATION**", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Hakan TEMELTAŞ** Istanbul Technical University	.....
Jury Members :	**Prof. Dr. Metin GÖKAŞAN** Istanbul Technical University	.....
	Prof. Dr. Işıl BOZMA Boğaziçi University	.....
	Doç. Dr. Turan SÖYLEMEZ Istanbul Technical University	.....
	Prof. Dr. Onur TOKER Fatih University	.....

Date of Submission : 25 September 2012
Date of Defense : 07 December 2012

To my family,

FOREWORD

There are many people I would like to thank for their time, patience, and support over the last several years. I have had a marvelous time at Istanbul Technical University, and I credit this fortune to my family, friends, and colleagues.

I would first like to thank my advisor, Prof. Dr. Hakan Temeltaş, who has been tremendously supportive of me during my time at Istanbul Technical University. Our discussions have played an integral part in developing my thesis work, and more generally, in discovering my research interests.

I am also thankful to Prof. Dr. Onur Toker for his great supports of me during my studies in the last few years. His advices are greatly appreciated. I am grateful to my fellow lab-mates, many of whom have become close friends. I have really enjoyed hanging out with them both in and outside of the lab.

My most valuable support by far has come from my lovely wife, Melek. I appreciate her patience, support, and understanding.

Most importantly, I owe an unlimited amount of gratitude to my family since they give me a tremendous amount of support over all these years. I am truly lucky to have such a close relationship with my aunt Sevinç. She has shown her love and support in an uncountable number of ways.

I also need to say sorry to my little baby Ahmet since I have to spend most of my time with studying instead of playing with him. I hope we will have great times after now on.

December 2012 Cihan Ulaş

TABLE OF CONTENTS

Page

ABBREVIATIONS

SLAM	**:** Simultaneous Localization and Mapping
ICP	**:** Iterative Closest Point
NDT	**:** Normal Distribution Transform
ML-NDT	**:** Multi-Layered Normal Distribution Transform
FbML-NDT	**:** Feature Based Multi-Layered Normal Distribution Transform
EKF	**:** Extended Kalman Filter
UKF	**:** Unscented Kalman Filter
CKF	**:** Cubature Kalman Filter
EIF	**:** Extended Information Filter
SEIF	**:** Sparse Extended Information Filter
RSPKF	**:** Randomized Sigma Point Kalman Filter

LIST OF TABLES

Page

LIST OF FIGURES

Page

3D SIMULTANEOUS LOCALIZATION AND MAPPING METHODS IN OUTDOOR AND LARGE-SCALE ENVIRONMENTS FOR AUTONOMOUS ROBOT NAVIGATION

SUMMARY

The problem of concurrent localization and mapping for a fully autonomous system in an unknown environment, where the external reference systems like Global Navigation Satellite System (GNSS) or Global Positioning System (GPS) is not available, is known as Simultaneous Localization and Map (SLAM) building problem. This problem has been one of the most popular research areas in mobile robotics in the last two decades, and important approaches have been proposed to solve this problem. The SLAM problem is considered as “chicken and egg” problem since both localization and mapping are highly related and cannot be solved individually for unknown and GPS-proof environments.

The traditional SLAM methods using Extended Kalman Filter (EKF) work without any problem in indoor environments. However, outdoor and large-scale SLAM is very problematic and there is no perfect solution implemented with actual systems. A group of SLAM methods uses features extracted from the high-volume data, which can be obtained from either Light Detection and Ranging (LiDAR) measurement system or stereo vision cameras. Another group of SLAM methods is based on scan matching algorithms, which provides rigid body transformation parameters of the robot motion. In this thesis, two groups of SLAM methods are investigated, and important contributions are made on both groups. First, a novel scan matching method, ML-NDT, based on normal distribution transform is proposed. This method uses a multi-layered structure with a new score function definition and provides fast and long-range measurement capability with respect to the conventional methods. Second, a new feature extraction method based on stochastic plane detection is proposed for outdoor and large-scale SLAM. This method is used in two ways; first, the extracted plane features are used with in ML-NDT scan registration algorithm to obtain more robust and fast scan matching method, and second the extracted features are used as landmarks for the feature based SLAM methods. In scan matching based SLAM algorithm, the Kalman filter (KF) is used to solve SLAM problem under the assumption of Gaussian noise, and in the feature based SLAM problem, traditional filters are used in the solution of SLAM problem with planar features. Finally, a data association algorithm based on the plane merging criteria is proposed for the planar-feature based SLAM. For the experimental results, actual LiDAR datasets obtained from the urban areas are used to show both scan matching and feature extraction methods performances. Finally, two benchmarking methods for 6D-SLAM is evaluated and used to compare the performance of the proposed feature based SLAM methods. The results show that the proposed methods can be used in outdoor and large-scale SLAM problem successfully.

OTONOM ROBOT NAVİGASYONU İÇİN DIŞ VE GENİŞ-ÖLÇEKLİ ORTAMLARDA 3D EŞ ZAMANLI KONUMLAMA VE HARİTALAMA YÖNTEMLERİ

ÖZET

Otonom araçların, Global Pozisyonlama Sistemi (GPS) gibi harici bir referans sistemi bilgisinin mevcut olmadığı, bilinmeyen bir ortamda ilerlerken aracın bir taraftan ortamın harita bilgisini çıkarırken diğer taraftan bu haritaya bakarak kendi konum bilgisini üretmesi Eş Zamanlı Konumlama ve Haritalama (EZKH) problemi olarak bilinir. Bu problem son yirmi yılda mobil robotik ve insansız araçlar alanındaki araştırmalarda en önemli çalışmalardan biri olmuştur. EZKH probleminde, konumlama ve haritalama birbirine bağımlı süreçler olduğundan ve ortamın haritasının önceden bilinmediği ya da GPS bilgisinin olmadığı durumlarda ayrı ayrı çözülemediğinden "tavuk-yumurta" problemi olarak da görülmektedir. Modelleme hataları, sensörler tarafından sağlanan bilgilerin belirsiz olması ve doğrusallaştırma problemlerinden dolayı EZKH probleminin çözümü genellikle deterministik yöntemler yerine olasılıksal yöntemlere dayandığı görülmektedir. Ancak olasılıksal yöntemler, haritadaki imleç sayısı ile karesel olarak artan bir hesaplama karmaşıklığına sahip olduğundan çözümü deterministik yöntemlere oranla daha zor ve karmaşıktır.

Önerilen bir olasılıksal EZKH yöntemin bilimsel bir sınıflandırma içerisinde nerede olduğu giriş kısmında detaylıca açıklanmıştır. Kısaca bahsetmek gerekirse bu tezde imleç (feature) ve hacimsel haritalama, iki grup EZKH yöntemi içinde kullanılmıştır. İlk olarak imleç tabanlı EZKH yöntemlerinde imleç çıkarma işlemi yapılırken, tarama eşleştirme tabanlı EZKH yöntemlerinde yoğun işlemlerin olduğu hacimsel veriler kullanılmıştır. EZKH probleminin sınıflandırılmasında diğer bir kavram da veri eşleştirmenin olup olmayacağı konusudur. Bu çalışmada yeni bir imleç çıkarma yöntemi önerildiğinden veri ilişkilendirme problemine de yeni bir yaklaşımla çözüm önerilmiştir. Sınıflandırma içerisinde bir diğer kabul de önerilen yöntemin çalışacağı ortam tipi ile ilgilidir. Bu tezde önerilen yöntemler kentsel bir alanda toplanmış olduğundan ortamın dinamik olduğu düşünülmektedir. Ayrıca önemli başlıklardan biri de çevrim kapama problemidir ki robotun görevini tamamladıktan sonra başlangıç noktasına geri geldiğini algılama problemi olarak tanımlanır. İmleç tabanlı EZKH probleminin çözümünde, önerilen imleç çıkarma ve buna yönelik veri eşleştirme yöntemi çevrim kapama işlemini otomatik olarak yaptığından yöntemin harici bir çevrim kapama yöntemine ihtiyacı yoktur. Ancak, tarama eşleştirmeye dayalı EZKH problemlerinin kararlı ve doğru çalışabilmesi için bir çevrim kapamaya ihtiyaç duyarlar. Bu tezde tarama eşleştirmeye yönelik yapılan çalışmalarda çevrim kapama problemi için herhangi bir bilimsel katkı yer almamaktadır.

Mevcut EZKH yöntemleri genellikle ortamın kapalı ve görece sınırlı bir alan olduğunu varsayarlar. Bu tip ortamlarda, Genişletilmiş Kalman Filtresi (GKF)

kullanan Geleneksel EZKH yöntemleri neredeyse hiç problemsiz çalışabilmektedir. Ancak, harici ve geniş ölçekli ortamlarda EZKH problemin çözümü daha zor ve gerçek zamanda kusursuz çalışabilen bir yöntem bulunmamaktadır.

Bir grup EZKH yöntemi, LiDAR veya stereo kameralar ile elde edilen hacimsel verilerden imleç çıkarma yöntemleri ile imleçler çıkarırlar. Dış ortamda imleç çıkarma sürecinin temel problemi, ortamın dinamik, yapısal olmayan ve önceden tahmin edilemeyen gürültü tipine sahip olmasıdır. Diğer bir grup EZKH yöntemleri de iki ardışıl taramadan robotun rölatif dönüşüm parametrelerini hesaplayan tarama eşleştirme algoritmalarına dayanır. Bu tezde, her iki grup EZKH yöntemi de irdelenmiş ve her iki alanda da önemli katkılarda bulunulmuştur.

İlk olarak, Normal Dağılım Dönüşümüne (NDD) dayanan yeni bir tarama eşleştirme yöntemi önerilmiştir. Bu yöntem çok katmanlı bir yapıyı yeni bir maliyet fonksiyonu kullandığından Çok Katmanlı Normal Dağılım Dönüşümü (ÇKNDD) olarak adlandırılmıştır ve klasik NDD' ye nazaran daha hızlı ve daha büyük miktardaki dönüşüm parametrelerini hesaplayabilme kabiliyetine sahiptir. Bu yöntemde, referans nokta bulutu öncelikle her katmanda farklı, aynı katmanda eşit hücre büyüklükleri bölünerek ayrıklaştırılır. Daha sonra her bir hücre içerisindeki noktaların ortalama vektörü ve kovaryans matrisi bulunarak nokta bulutunun Normal Dağılım Dönüşümü (NDD) bulunmuş olur. Eşleştirme sürecinde ise eşleştirme yapılacak olan ikinci nokta bulutundan seçilen noktaların rölatif dönüşüm parametreleri ile modellenmiş referans nokta bulutu üzerine izdüşümü ile hangi hücreye karşılık geldiği bulunur ve bu nokta ile hücre ortalama ve kovaryans değerlerine bağlı olarak bir hata vektörü tanımlanır. Bu hata vektörünün tüm noktaları kapsayan toplamı Newton yöntemi ile iteratif bir şekilde minimize edilerek rölatif öteleme ve dönmeyi birlikte ifade eden dönüşüm parametreleri hesaplanır.

İkinci bir katkı olarak, harici ve geniş ölçek ortamlarda EZKH problemine yönelik olarak olasılıksal bir algoritma ile düzlem algılamaya dayanan yeni bir imleç çıkarma yöntemi önerilmiştir. Bu imleç çıkarma yönteminde olasılıksal düzlem algılama yöntemi önerildiğinden yöntem gürültü ve dış ortamda daha dayanıklı sonuçlar vermektedir. Bu yöntemde, nokta bulutu öncelikle eşit hacimli hücrelere bölünür ve her bir hücredeki noktalara varsa bir düzlem uydurulur. Daha sonra bu düzlemler iki boyutlu ana eksenleri üzerine izdüşürülürler ve izdüşümleri üzerinden düzlemlerin dışbükey noktaları bulunur ve bu noktalar tekrar üç boyuta geri izdüşümleri gerçekleştirilir. Son olarak, dolu hücredeki düzlem parçacıkları bu çalışmada önerilen bir düzlem birleştirme yöntemi ile birleştirilerek daha büyük boyutlu düzlemler ede edilir. Bu imleç çıkarma yönteminden iki şekilde faydalanılmıştır. Birincisi çıkarılan düzlemsel imleçler ÇKNDD tarama eşleştirme yöntemin kullanılarak daha dayanıklı ve hızlı eşleştirme yapabilen imleç tabanlı yeni bir tarama eşleştirme yöntemi önerilmiştir. İkincisi ise çıkarılan bu düzlemsel imleçler imleç tabanlı EZKH probleminin çözümünde "işaretçi (landmark)" olarak kullanılmıştır.

Son olarak, önerilen tarama eşleştirme ve imleç çıkarma yöntemlerinin EZKH problemine nasıl uygulanacağı sunulmuştur. Tarama eşleştirme yöntemi, ÇKNDD, Kalman Filtresi (KF) ile uyumlu hale getirilmiş ve EZKH problemin çözümünde kullanılmıştır. İmleç çıkarma yöntemi de literatürde önemli yeri olan Genişletilmiş KF (GKF), Kübik KF (KKF) gibi filtrelere uyarlanmış ve geçerliliği gösterilmiştir. Bir sonraki aşamada yeni bir filtre olan Rassallaştırılmış Sigma Nokta KF (RSNKF) EZKH problemine uyarlanmış ve düzlemsel imleçler ile deneyler gerçekleştirilmiştir.

Ancak bahsedilen filtre yöntemleri, durum vektörünün boyutunun artması ile hesaplama karmaşıklığı arasında karesel bir ilişkinin olmasından dolayı geniş ölçekli haritalamada önemli bir dezavantaja sahiplerdir. Nihai olarak bu tezde bu amaçla, düzlemsel imleçleri kullanan, geniş ve harici ortamlarda EZKH probleminin çözümü için, imleç sayısından bağımsız hesaplama yapabilen Seyrek Genişletilmiş Bilgi Filtreleri (SGBF) önerilmiştir. SGBF' leri aslında Kalman filtrelerinin bir kanonik formu olan bilgi filtrelerinin seyreltilmesi ile oluşmaktadır. Robot ile pasif imleçler arasındaki zayıf bağa sahip olan bağlantılar ortadan kaldırılarak bilgi matrisinin seyreltilmesi gerçekleştirilir. Bu sayede bilgi filtrelerinin önemli bir dezavantajı olan ölçüm güncelleme imleç sayısından bağımsız olarak sabit bir zaman içerisinde çözümlenebilmektedir. Önerilen düzlemsel imleçler noktasal imleçlerden farklı olduğundan olasılıksal veri ilişkilendirme yöntemleri kullanılamaz; yerine imleç çıkarma yönteminden elde edilen düzlemlerin diğer özellikleri kullanılarak ilişkilendirme yapılır. Bu sayede bilgi filtrelerinin darboğazı olan veri ilişkilendirme problemi de çözümlenmiş olur.

Deneysel sonuçlar şehir ortamında lazer tarayıcılar kullanılarak toplanan gerçek bir veri seti üzerinden gerçekleştirilmiştir. Bazı durumlarda yöntem iyileştirme çalışmaları için simülasyon deneyleri de yapılmış ve sonuçlar tezin ilgili bölümlerinde yer verilmiştir. Son olarak EZKH yöntemlerinin değerlendirilebilmesi amacıyla literatüre mevcut olan iki yöntemden faydalanılmış ve imleç tabanlı tüm filtre performansları karşılaştırılmıştır. Sonuçlar önerilen tarama eşleştirme ve imleç çıkarma yöntemlerinin dış ortamda ve geniş ölçekli EZKH probleminin çözümünde başarılı bir şekilde kullanılabileceğini göstermektedir.

1. INTRODUCTION

The navigation of mobile robots and the unmanned vehicles is a general concept of the methods considering the environment conditions and avoiding from the obstacles to be able to move from a starting point to the target point. Moreover, in autonomous navigation, the unmanned vehicles eliminate the human oversight while the robot is fulfilling the navigation tasks. In the literature, there have been important scientific contributions for robot self-localization in the past two decades. In general, two important throughputs are required for autonomous navigation. The first one is that the information of the map where the robot travels, and the second one is that the information of the robot position and orientation, called localization. On the one hand, robot extracts or learns the map of the environment; on the other hand, it computes its own localization information based on the extracted map while moving in an environment in which the map of the environment is unknown. This “chicken and egg” problem is known as the “Holy-Grail” problem in the autonomous robotic society and named as Simultaneous Localization and Mapping (SLAM) problem.

In the literature, it is possible to see probabilistic methods rather than deterministic methods in the solution of the SLAM problem. The reason of this is that the information coming from the sensors are not exact and the uncertainty factors due to modeling errors and noises in the systems. By considering these effects, it can be said that the probabilistic methods are more robust than the deterministic methods. However, the growth of computation time for solving the problem is an important issue for probabilistic methods. Especially, when the size of the map increases, the computation times increases quadratically. This is a significant problem of probabilistic methods since SLAM is applied to the real-time systems. It has been proposed various methods to solve this problem in literature, and they are discussed in later of this chapter.

The solution of probabilistic SLAM methods is inspected in under a certain taxonomy, which is accepted by the robotic community. Any method proposing a solution to SLAM problem has to specify its class in this taxonomy. The SLAM

taxonomy and the place of proposed thesis in this category are summarized as follows:

- **Classification in terms of mapping:** It can be volumetric or feature-based. The volumetric maps allow us to obtain the map sensitively, but it is not preferred in real-time solutions due to high dimensional data storage and processing time. Instead, the features are extracted from the volumetric data by applying signal processing techniques, and these features are used for SLAM. *In this thesis, both volumetric and feature based maps are used.*
- **Classification in terms of data association:** Data association is also known as the correspondence decision where it asks the "yes or no" question to the observed feature: Did I observe this feature before? Most of the probabilistic SLAM studies do not deal with the data association problem and assume that it is already solved. *In this thesis, a data association method is proposed for the feature extraction methods.*
- **Static or Dynamic Environment:** While static SLAM methods assume that the environment is static, dynamic SLAM methods allow for changes. *In this thesis, the environment is considered as dynamic since the experimental datasets obtained in urban areas are used.*
- **Loop Closure:** It comprises that an autonomous vehicle can come to its starting point and can detect it after the long-range task. *This is similar to the data association problem, and the solution is comprised by this thesis.*
- **Active or Passive SLAM:** In passive algorithms, the SLAM algorithm only observes and generates the localization information, and the robot is controlled by another algorithm. On the other hand, in active SLAM, the robot actively explores the environment to be able to gather accurate map while restricting the robot motion. *This thesis is in the class of passive SLAM.*
- **Single-Robot or Multi-Robot SLAM:** While single-robot SLAM is based on one robot, the Multi-robot SLAM, including more than one robot, exploration gains more popularity in the SLAM community. *In this thesis, the SLAM methods are proposed for a single-robot.*

1.1 Problem Statement and Motivation

The existing methods for solving Simultaneous Localization and Mapping (SLAM) problem assume that the environment is indoor and relatively small area. In this type of environments, the conventional SLAM methods using Extended Kalman Filters (EKF) works almost perfectly. However, SLAM in outdoor and large-scale environment is not as easy as in indoor, and it is still an open problem in the literature. The main difficulty for outdoor SLAM is that the feature extraction is very problematic due to unstructured, dynamic, and noisy environment. Therefore, scan matching methods play an important role in 3D outdoor SLAM since they do not depend on the feature extraction. However, due to advantages of compact map representation and improved mathematical background in feature based SLAM; the researchers are looking for robust and fast feature extraction methods for outdoor and large-scale SLAM. In addition, the traditional scan matching methods have several drawbacks such as high data storage, long registration times, and loop closure problem. First, the scan matching problem and its application to SLAM is explained, and secondly the feature extraction methods and feature based SLAM methods are investigated.

1.1.1 Scan Matching and Volumetric SLAM

Scan matching is mostly applied to Light Detection and Ranging (LiDAR) systems, which is an optical remote sensing technology measuring the distance to an object by using laser beam and generates a point cloud of the observed scene with a range and bearing information. In volumetric SLAM, scan matching is used to find the relative transformation parameters of the robot motion by aligning two consecutive scans. Then these relative transformations generate a network where the robot poses are considered as the nodes of this network and the relative transformation parameters are the links. The solution of this network is obtained by formulating a procedure based on the maximum likelihood criterion to combine these spatial relations optimally [31]. There are two fundamental scan matching methods used by the researchers, which are iterative closest point (ICP) [2] and Normal Distribution Transform (NDT) [3].

Conventional ICP method is based on minimization of the sum of all point-to-point distances. There are also point-to-plane and plane-to-plane variants of ICP to

increase integrity. An important disadvantage of an ICP algorithm is the requirement of a nearest neighborhood searching (NNS) algorithm, which is the most time consuming part of the algorithm, to find correspondences between two scan points. In addition, ICP does not release any surface structure information; therefore, it is considered as sensitive to noises and outliers. Another issue is the data storage necessity, especially for large maps, since the raw 3D point clouds have to be saved. Therefore; the ICP seems inappropriate for large scale and outdoor SLAM; thus, another method, Normal Distributions Transform (NDT), is proposed as an alternative to ICP.

Normal Distribution Transform (NDT) is initially applied to 2D data by Biber and Strasser and extended to 3D data in [54]. The main contribution in NDT is that the point cloud is divided into regular cells, and the surface is represented by the probability density functions of each cell. Therefore, the map is represented in a much more compact form than ICP. This provides fast scan matching performance and less memory storage for large maps. Another advantage of the NDT over ICP is that it helps to solve the detection of loop closure problem due to it is compact form of the surface by using appearance map based SLAM techniques [32].

1.1.2 Feature Extraction and Feature Based SLAM

The second group of SLAM methods use landmarks or features in a probabilistic framework to solve SLAM problem [14]. The landmarks can be either artificial objects like reflectors or features which are extracted from the point cloud [39]. The artificial objects based methods are restricted since they can solely be applied to the well-known environments and must be distributed correctly in the environment. Therefore, the feature extraction methods are proposed to find geometric features such as lines [5], corners [75], and curves [40] in 2D and planes in 3D [47]. Then these landmarks and the marginalized robot pose are combined into a state vector and estimated statistically by using traditional filtering methods such as Kalman Filter, Information Filter, and Particle Filters [14]. The main aim here is to estimate the robot pose and landmark positions as possible as small error uncertainty.

In feature based probabilistic SLAM, the most significant part is to find robust features. These features must be represented in a very compact form since the state vector is composed of these landmarks. The size of the state vector is a critical issue

if the number of features exceeds a few hundreds due to the difficulties on taking the inverse of large matrices. The feature based SLAM is applied to indoor and 2D case successfully recently [30], [7]; however, large-scale and outdoor SLAM in 3D is still an open issue. In [38], an appearance-based method for augmenting maps of outdoor urban environments is presented. The feature vector is compiled via combining both 3D geometrical features obtained from laser data and appearance features obtained from camera with semantic labels. The existing feature extraction methods work in mostly structured environment. For unstructured or semi-structured complex environments like urban areas, a more robust and fast feature detection algorithm is needed.

Another prominent part of a feature based SLAM algorithms is the filtering method used for solving the localization and mapping problem. The most influential filtering method in SLAM is the Extended Kalman Filter (EKF), and EKF based SLAM assumes that the robot motions and observations involve Gaussian noises. In addition, the variance of the noise in the posterior should be relatively small for tolerable linearization errors. For high-level noises and aggressive non-linear model functions, Unscented Kalman Filter (UKF) approximates the true state vector and covariance matrix better than EKF by applying the unscented transform [23]. Recently, another filtering method Cubature Kalman Filter (CKF) has been proposed instead of UKF for nonlinear state estimation under Gaussian noise by Arasaratnam and Haykin [1], where it is shown that CKF provides more accurate filtering results than the existing Gaussian filters and solves large spectrum of nonlinear problems. CKF is applied to point-feature based SLAM in 2D in a simulation environment [43]; thus, more practical applications are necessary to show its validity for 6D-SLAM.

The ultimate purpose of this thesis is to propose a SLAM method working long-time in a large-scale and outdoor environment; therefore, the traditional EKF, UKF, and CKF cannot be used with the classical feature representation due to large covariance matrix, which grows constantly with new observations. There are two well-known filtering methods for solving SLAM problem, which are Extended Information Filters (EIF) and Particle Filter (PF). Since the EIF is only a canonical representation of the covariance form or Kalman form, it also has the same quadratic time complexity problem with EKF. PF is another filtering method in solving SLAM

problem and has the important difficulty of matching of samples to the target probability distribution. Since this issue becomes more dominant with the augmentation of the size of the map, it is not suitable for long-range tasks. Thus, EIF and PF are not appropriate for the large-scale SLAM and for this thesis, too. However, it is shown that while the normalized covariance matrix in Kalman representation is not sparse, the normalized information matrix is sparse in nature. In addition, the measurement update can be achieved in constant time since the update equations are in additional form. However, in the marginalization of the motion update step, the information matrix loses the sparseness property and so does the constant time solution. A solution is proposed by Thrun by introducing Sparse Extended Information Filter (SEIF) [58] which is based on deactivation of the weak links between robot and passive features.

The last but not least issue in feature based SLAM is the data association problem. The data association is a fundamental problem for point features in probabilistic SLAM. The point features might be seem different or sometimes may not be even seem from different viewpoints. The data association algorithm must decide on the correct correspondence, and so the distance measurement between two features is a crucial issue. The classical point feature based methods minimizes the conventional Mahalanobis distance function for solving the correspondence problem. The wrong data association may result in high errors or even diverge on estimations. Therefore, more sophisticated feature extraction and data association methods are required for robust outdoor and large-scale 6D-SLAM.

1.2 Historical Review of Related Studies

The probabilistic SLAM problem is firstly brought forward in 1986 IEEE Robotics and Automation Conference held in San Francisco [14]. Researchers had been investigating solutions for mapping and localization by using estimation theory. Durrant-Whyte [13] proposed a statistical method describing relationships between landmarks using uncertain geometry in 1987. In this paper, he shows that there is high degree of correlation between estimated landmarks positions, and these correlations increases with the new observations. It is shown that the reason of the correlation is due to the common error in the estimated robot location [28]. This means that a complete solution to the localization and mapping problem requires a

joint state composed of the robot pose and landmark positions, which are updated with the observation of each landmark. This study is not interested in the convergence problem of the map and assumed that the estimated map errors do not converge and perform irregular behavior with unlimited error growth.

A convergent solution was realized when the mapping and localization problem were combined into a single estimation problem, called as Simultaneous Localization and Mapping problem (SLAM) [15]. This framework brings to light a significant results that the more accurate solution is obtained when the correlations grows, which was known conversely by then. The SLAM theory and many of the initial results were developed by Dissanayake et al. [11], and there are important groups working on SLAM problem, which cannot be passed without mentioning, at MIT [29], Zarogaza [9], the ACFR at Sydney [20] in the applications of indoor, outdoor, and sub-sea environments.

The main focus of this thesis is in the area of large-scale and outdoor SLAM. In this regard, two different categories, which are scan matching and features extraction methods, for solving the large-scale and outdoor concurrent localization and mapping problem in 6D is extensively investigated in the next chapter.

1.3 Contributions

The contributions provided by this thesis are given in the following subsections.

1.3.1 A Novel Scan Matching Method

An improved version of the Normal Distribution Transform (NDT) is proposed for scan matching of two 3D point clouds (PCs). The key point in NDT is to use fixed cell sizes to discretize the PC. While the usage of large cell sizes makes the algorithm fast, the convergence area becomes large, too. On the other hand, the usage of small cell sizes makes the process slower, but the result may become more accurate if an adequate initial relative pose estimate exists. Multi-Layered NDT (ML-NDT) method assigning the cell sizes automatically based on the point cloud boundaries is proposed, and then the point cloud is divided into q^i equal-sized cells in the i^{th} layer. Here i represents the layer index and q represents the division constant in this layer. The NDT is applied to each layer, and the probability density functions of each cell are stored for scan registration. In registration step, the algorithm starts

with the first layer having the largest cells and switches to the lower layers when the number of iteration reaches to a threshold value.

In conventional NDT, the score function is chosen as Gaussian probability function, whereas the score function is chosen as Mahalanobis distance in ML-NDT. In addition, Newton and Levenberg-Marquardt optimization methods are integrated to the proposed method, respectively. Another important powerful part of the ML-NDT is to work without using shifted cells as in NDT, which is used to minimize the discretization error with the cost of five times the original computation time [34]. Finally, we discuss the effect of subsampling strategies in the performance evaluations of the NDT and ML-NDT. Since there is no study proposing a sophisticated sampling method for NDT so far, this study will also be a leading study in this area. As a result, an improved scan matching method having fast and long-range measurement capability is proposed.

1.3.2 A Feature Extraction Method Based on Probabilistic Plane Detection

While in 2D mostly line, corner, and single points are used as features, in 3D the favorite feature can be thought as the plane. The plane extraction algorithms works well in indoor and structured environments, but they have problems in outdoor and especially in complex environments, where buildings, trees, and moving objects exist.

As a second contribution of this thesis, a robust feature extraction algorithm that is particularly dedicated to outdoor and large-scale SLAM methods is proposed. In this method, robust planar features are extracted as in [71] for structured and indoor environment; however, the proposed method in this thesis works also for the unstructured and outdoor environments. The method is applied in three steps. In the first step, also known as the discretization or splitting step, the point cloud is divided into regular cells. A fixed cell size is chosen based on the point cloud distribution and each point is placed to the corresponding cell. Then the plane segments of each cell are found by using Random Sample Consensus (RANSAC) algorithm [16]. Since RANSAC algorithm is very robust to outliers, it is much more suitable than classic plane fitting methods for outdoor applications. After extracting the plane segments, they are projected to *x-y* plane using Principal Component Analysis (PCA), and then their convex hulls (or minimum rectangular bounding boxes) are found to represent

the planes in a finite form. Then these segments are merged by applying a plane-merging algorithm. Weingarten and Siegwart uses two criteria in the merging step for indoor implementation, but these are not sufficient for the outdoor case since the environment is not structured and contains noisy data [71]. Therefore, we add one extra condition based on pooled covariance matrix to test the closeness of two planes in complex environments. Finally, if a merged plane satisfies the area and number of point's criteria conditions, they are considered as landmarks for feature based SLAM. The data association based on the semantic data provided by the proposed method, and the applicability of the proposed method to SLAM problem are also discussed in the related section.

The proposed feature extraction method is used for two different SLAM problems. First, the extracted planar-features are used for robust and fast scan matching. Second, the extracted planar-features are used in feature based SLAM.

1.3.3 A Fast and Robust Feature based Scan Matching Method

The third contribution is a fast and robust scan registration method using planar features extracted from the PCs for 6D outdoor SLAM. The method is composed of two main parts which are feature extraction and scan matching. In feature extraction, a probabilistic plane extraction procedure is applied via RANSAC algorithm to the discretized point cloud. As a scan matching method, an improved version of the NDT called Multi-Layered NDT (ML-NDT) is used. Concisely, the feature extraction and scan matching methods, explained in the previous sections, are combined for fast and robust scan matching method.

In generative phase of the algorithm, probability density functions (pdfs) of each scan in all layers are computed for the reference scan. For the input scan, which is the one to be registered to the reference scan, the plane inlier points are extracted with the stochastic plane detection method. In matching process, instead of using the all scan points (or sampled points) as in original NDT, or ML-NDT, these inlier points are used. By doing so, the Feature based ML- NDT (FbML-NDT) algorithm is possessed of several advantages. The first one is that the registration processing time, which is the most time consuming part of the scan matching algorithms, is highly reduced. The scan matching method is more robust than the conventional methods since the matching is based on plane structures, which provides more consistent .

In this thesis, the effect of sampling strategies like random sampling and grid based sampling on the performance of NDT, ML-NDT are investigated, and the proposed feature extraction method is considered as special sampling method and compared to these subsampling methods.

1.3.4 Scan Matching based SLAM with Kalman Filters

In this contribution, the ML-NDT and Kalman filters are combined to solve 6D robot localization problem. To be able to implement a Kalman filter based SLAM, one has to know the state space representation of the controlled system and observation models. This scan matching and Kalman filter based SLAM method is an extension of the study on localization for rescue robots based on NDT scan matching and Kalman filter [22].

1.3.5 Planar-Feature Based SLAM

The representation of planar-features in state vector is different from the point features representation. While the positions of features are held in three dimensions in the state vector for point features, for planar-features, the infinite plane parameters are held in four dimensions in the state vector, which are normal vector in three dimensions and a plane minimum distance parameter to the origin of global frame.

First, the proposed planar-features are used with the Extended Kalman Filer for outdoor 6D-SLAM since it is the de facto standard filtering method in SLAM, and it is used as a reference filter in the comparisons. Then the planar-features are applied to more sophisticated filtering methods such as Unscented Kalman Filters (UKF), Cubature Kalman Filter (CKF), and Randomized Unscented Kalman Filter [12], which is a quite new filter and has not been used in SLAM before. Finally, SEIF, a scalable filtering method, provides a constant time solution for large-scale SLAM problems [58] and is used with the planar-features proposed in this thesis.

1.3.6 Semantic Data Association for Planar-Features

The most of the data association methods are proposed for point-features and they are based on the minimization of the conventional Mahalanobis distance function. Since this thesis proposes a novel feature type, planar feature, the classical data association methods cannot be used. The reason is that the planar-features are represented as a infinite form in the state vector. Instead, extracted feature properties

like plane convex-hull, plane area, plane covariance matrix, and plane center point is used. These enriched properties of the planar-features are considered as semantic data and the data association method is called as “Semantic Data Association”.

Next section outlines the organization of this thesis.

1.4 Thesis Outline

Following the introductory part in this chapter, 6D SLAM methods based on LiDAR systems are explained in Chapter 2, where more details on point cloud generation methods, scan matching, feature extraction, and their application to SLAM problem are given.

In Chapter 3, the proposed scan matching algorithms and their application to SLAM problem is elucidated. In this section, two related scan matching method is proposed. First, ML-NDT, an improved version of the normal distribution transform (NDT) using multi-layered structure and different score function, is explained. Second, the feature extraction method and a new feature based scan matching method called FbML-NDT, which is faster and more robust than ML-NDT, is presented.

The proposed feature extraction method in Chapter 3 is adapted to the feature based 6D-SLAM in Chapter 4. The theory of the planar-feature extraction and its application to the SLAM problem is explained in detail. The usage of the planar-features with the following filters EKF, UKF, CKF, RSPKF, and SEIF is given in this chapter.

Experimental results and the performance evaluations are given in Chapter 5 and in Chapter 6, two benchmarking methods for the proposed planar-feature based 6D-SLAM methods are utilized, and the results are discussed for relatively large dataset. Finally, Chapter 7 provides the concluding remarks including summary and future directions.

2. LIDAR BASED 6D SIMULTANEOUS LOCALIZATION AND MAPPING METHODS

In LiDAR based Simultaneous Localization and Mapping (SLAM), the initial step is to generate the point clouds (PCs) or 3D data of the environment in Euclidean space. For this reason, firstly, the PC generation methods are explained in the next subsection. Then the two groups of SLAM methods using different mapping structures, which are based on scan matching and feature extraction, in the literature are discussed.

2.1 3D Point Cloud Generation Methods

2D laser range finders are commonly used for obtaining plane structure of the environment. Although there are commercial 3D laser range finders in the marketplace, since their high cost and restricted flexibility, they are not used by the researchers. Instead, the third dimension is obtained by rotating one or two 2D laser scanners in a particular axis depending on the application type as shown in Figure 2.1. In the literature, there are various 3D laser-scanning methods depending on the configuration of the actuator, which is mostly a servo drive or a step motor, and 2D scanners. These methods allow different scan planes and rotation axis leading to different views. Moreover, high measurement densities close to an axis are obtained by rotating around this axis. The density propagation may become intensive for a certain area, and this may lead to a misuse of the scanning ability.

The scanning time in three dimensions mostly takes more than 5 seconds and the memory storage can reach to several gigabytes even for small areas. Fast 3D scanning methods and data acquisition systems are discussed in [72]. In this study, three basic combinations of the servo drive and 2D laser scanner are investigated to gather 3D scanning data or point cloud as shown in Figure 2.2. Roll scan mode can be used for safety applications, and pitch scan can be used for object recognition applications. On the other hand, yaw scan can be used for outdoor implementations.

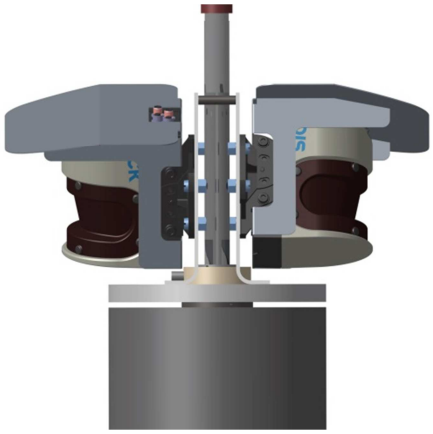


Figure 2.1 : 3D LiDAR measurement system consisting of two 2D laser range sensor and a servo drive.

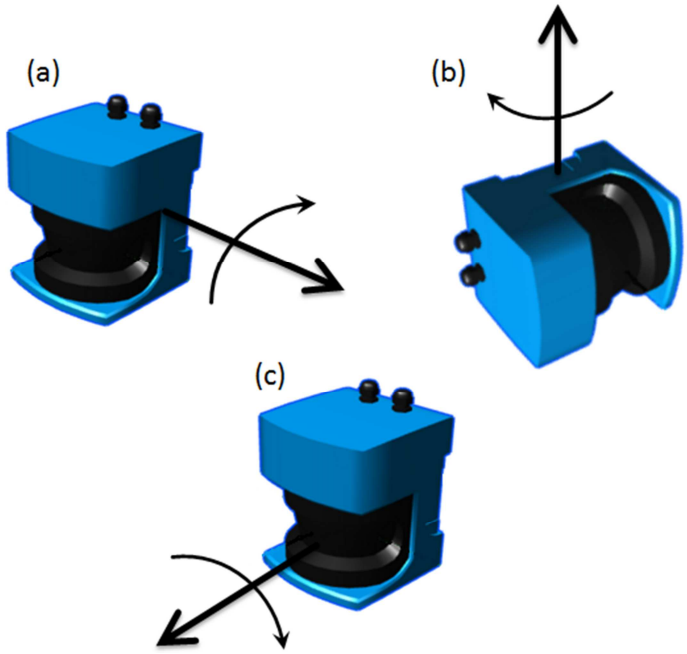


Figure 2.2 : Scanning methods: (a)Pitch scan. (b)Roll scan. (c)Yaw scan.

The distribution of the point cloud in LiDAR systems is not homogeneous as it is in camera based systems. The measurement density is maximum for beams which parallel to the rotation axis, and it is minimum for orthogonal to rotation axis. In Figure 2.3, the 2D scanner is rotated from -90 to +90 degrees with 0.2 degree steps around the pitch axis (*y*-axis), and it is rotated from -90 to 90 degrees around the roll axis (*x*-axis) in Figure 2.4. As seen from the figures high density occurs on edges with respect to scanner for the pitch scan mode, on the other hand high density occurs only in front of the 2D scanner for the roll scan mode.

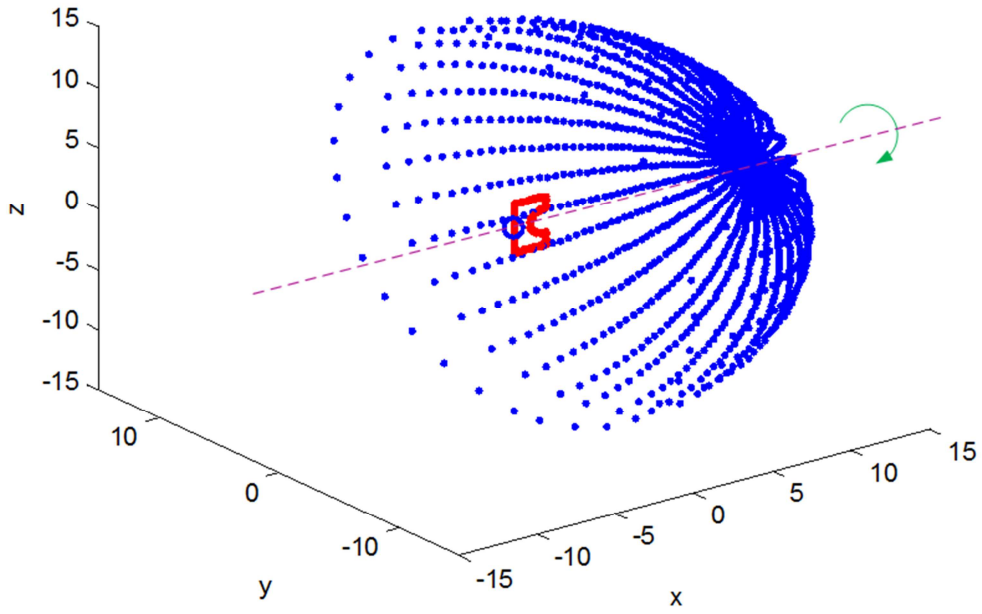


Figure 2.3 : Point cloud distribution of the roll scan.

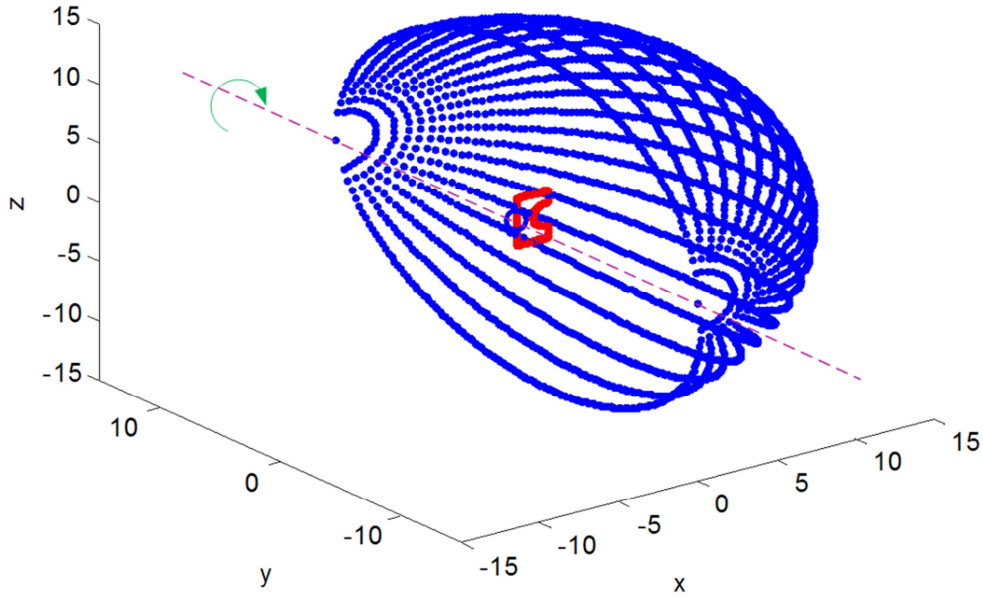


Figure 2.4 : Point cloud distribution of the pitch scan.

According to this information, it can be designed a custom hardware by using two scanners mounted back to back with 90-degree angle for outdoor and large-scale applications as shown in Figure 2.5. A commercial scanning system including two 2D scanners mounted back to back with 90 degree angle is designed by Fraunhofer IAIS [17]. Based on this set up, the 3D point cloud is more homogenous than the other setups as shown in Figure 2.5.

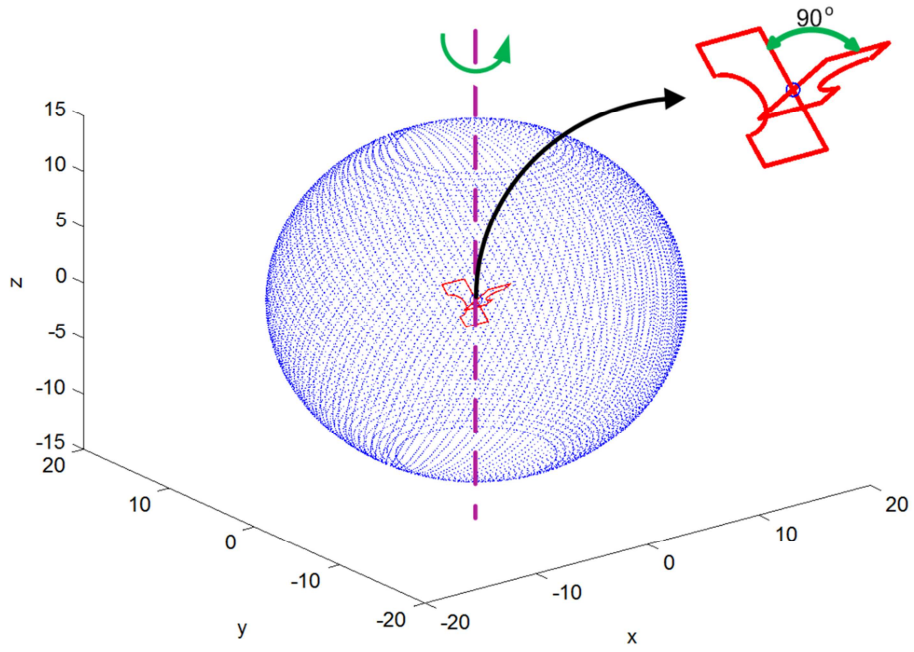


Figure 2.5 : Point cloud distribution of the custom design.

2.2 Scan Matching Methods and Dense Mapping

After point clouds are generated, they are geometrically aligned with a scan matching algorithm to obtain relative transformation parameters. Then these relative transformation parameters are put into a network to solve SLAM problem with graph realization methods [31]. Iterative Closest Point (ICP) and Normal Distribution Transform (NDT) are well-known methods for 2D and 3D scan matching [2], and the next section introduce the ICP scan matching method.

2.2.1 Iterative Closest Point (ICP)

The conventional ICP is based on point-to-point sum of distance minimization; therefore, it requires high computation time on nearest neighborhood search (NNS) for finding corresponding points. Typically, the NNS takes *O[MN]* computation time, and this time can be reduced up to *O[Nlog(M)]* by using kd-tree space quantization method [19]. On the other hand, there are important improvements on ICP such as point-to-plane, plane-to-plane, and their generalized variant proposed by Segal et al. [52]. The main idea in Generalized ICP is to use planar approximations to apply a plane-to-plane minimization by utilizing the structure of the environment. Moreover, a visually bootstrapped method using generalized ICP method is proposed by Pandey [45]. In order to improve the performance of the ICP, there are some

methods using geometric metric like Metric based ICP (MbICP) in 2D and its generalized version in 3D [29]. In [50], the efficient ICP variants and sampling strategies are discussed. They enumerate and classify many of these variants and evaluate their effect on the speed and accuracy. They also propose a new high-speed method by combining the ICP variants. Recently, an alternative method to ICP called Normal Distribution Transform is introduced.

2.2.2 Normal Distribution Transform (NDT)

Normal Distribution Transform (NDT) was introduced by Biber and Strasser [3] for the 2D scan matching. The main idea in NDT is to represent the point cloud as the sum of normal distributions. This is achieved by dividing the surface into regular cells and calculating the mean vectors and covariance matrices of each cell. Since the 3D point cloud maps require large amount of memory space, the surface of the point cloud should be represented in a compact form. This form should hold important surface characteristics such as orientation and smoothness. ICP method does not include any information about the surface structure. Therefore, the NDT can be considered as a powerful alternative method to ICP for 3D scan registration in outdoor. NDT has become very popular due to its faster convergence time, compact map representation, and appropriateness of significant SLAM problems such as loop closure [32]. The main advantage of the NDT over ICP is that there is no need to compare each point to find the correspondences. Thus, the most time consuming phase of ICP, the NNS, is not necessary in NDT. Instead, the reference scan is first discretized (divided into regular cells) and probability density functions (pdfs) of each cell is computed. Then a score function is optimized by using Newton optimization method for each point of the input scan. Another important advantage of NDT is that the memory storage is very less than ICP because NDT only stores the mean and covariance information while ICP has to store all point clouds for mapping. Consequently, NDT based SLAM methods can be considered as more suitable than ICP based methods for large scale and outdoor environments. The ICP and NDT algorithms are comprehensively compared in terms of their accuracy, success rate, and the convergence speed in the paper of Magnusson et al. [34]. The paper concludes that NDT can converge from a larger range of initial pose estimates than ICP and perform faster. Recently, there have been important improvements in traditional NDT to increase accuracy and speed. A marker free registration is adapted

to 3D laser scan by dividing the point cloud into multiple slices and applying 2D NDT algorithm [10]. They proposed three modifications on the original NDT: a rough to fine strategy, multiple slices, and Levenberg-Marquardt algorithm as optimization method. In [54], an improved 3D NDT for 3D scan matching is proposed. The main contribution of this study is to modify the 2D NDT to 3D NDT. They used two different cell sizes for dual resolution to accelerate the method. They start with large cell size, and after the convergence of the score function, they use small cell sizes. An interesting convergence calculation method of the NDT based high resolution grid map for 2D NDT is introduced in [26]. They used small cell sizes for high-resolution grid map representation and applied eigenvalue expansion to speed up the convergence time. A similar representation is also used for sub-grid object recognition by Takubo [55].

2.2.3 Data Sampling Methods for Fast and Efficient Scan Matching

Selection of a subset from all points in point cloud is a critical issue for a fast and robust scan matching. In general, there are several sampling strategies implemented with ICP [50] and can be categorized as follows;

- Always using all points
- Uniform subsampling
- Random subsampling in each iteration
- Selection of points with high intensity gradient
- Selection of points from the one scan or both scan
- Selection points such that the distribution of normals among selected points is as large as possible.

From the SLAM with NDT point of view, if the all methods are considered, we can conclude that the first method is not suitable for two reasons. The obvious one is that the number of points to be matched might be so much; therefore, the convergence time may take so long. The second reason is that the point cloud might be very noisy or unstructured due to the nature of the environment or the uncertainty of sensing devices. Therefore, a sampling strategy must be carried out. The second sampling strategy, which is uniform subsampling, is also may not be suitable if the point cloud is not uniformly distributed, which is mostly encountered in 3D outdoor scanning due to hardware configuration. The third strategy, random subsampling in each iteration, seems reasonable, but again due to non-uniform distribution the dense

regions will have more points. Since we do not have any intensity information in LiDAR only systems, the fourth strategy, selection of points with high intensity gradient, cannot be considered for NDT. The fifth method, selection of points from both scan, can be applied; in addition, since the point cloud is represented by sum of Gaussians, in NDT generative process, the sampling should be dense enough for good modeling. For that reason, we recommend very dense sampling for the reference scan. Finally, the last strategy is also meaningful for the input scan; however, finding the large normal distributions for selected points is not easy for outdoor 3D case and requires extra computations.

Grid Based Sampling Strategy In addition to mentioned sampling methods, the grid based sampling (GBS) strategy is proposed without any extra computational cost. In GBS, the reference point cloud is first digitized by dividing the point cloud into fixed cells as in the ML-NDT splitting process. Then the equal-number of points are sampled randomly from each cell of the input scan. With this way, the distribution of the sampled point cloud becomes more homogeneous than the convention random sampling. The splitting process of ML-NDT algorithm and the discretization process of the Grid Based Sampling (GBS) strategy are the common procedures. The only difference is that they are applied to the different scans. If we look at the ML-NDT algorithm given in Table 1, the reference scan is first discretized (splitting process), and then the mean vectors and the covariance matrices are computed for each cell of all layers (generative process). In the registration step of the algorithm, the points are sampled from the input scan and registered on the reference scan. The GBS strategy is based on the same splitting process, but it is applied to the input scan. However, if we consider the consecutive nature of the problem, the input scan in time t_i will be the reference scan in time t_{i+1}. For that reason, the splitting process of the ML-NDT is skipped in time t_{i+1} since it is already obtained in time t_i. Obviously, this is not true for the time t_o, and the two scans are discretized at the beginning of the robot motion. However, this initial case can be ignored if we consider the overall robot motion process.

In Figure 2.6(a), the point cloud is subsampled randomly. As it is seen, the regions close to the origin is highly emphasized. There are totally 18738 points in this illustration, and it is almost equal to 25% of the original data. On the other hand, when the grid based sampling (GBS) is applied; the Figure 2.6(b) is obtained. The

cell size is chosen as 1 meter and if the number of points in a cell is more than five, only five points are picked from this cell randomly. If there are less than five and greater than two points, all points are taken. In this figure, there are totally 3396 cells containing 16619 points. It is almost equal to 22% of the original data. It is possible to conclude that the point distribution becomes uniform when GBS is used. Finally, each part of the point cloud has the equal weights in registration. If the sensor measurements have large errors on distant regions, the weights of the distant cells to the origin can be reduced, but we did not consider this case in our experiments and behave equal to each cell.

(a)

(b)

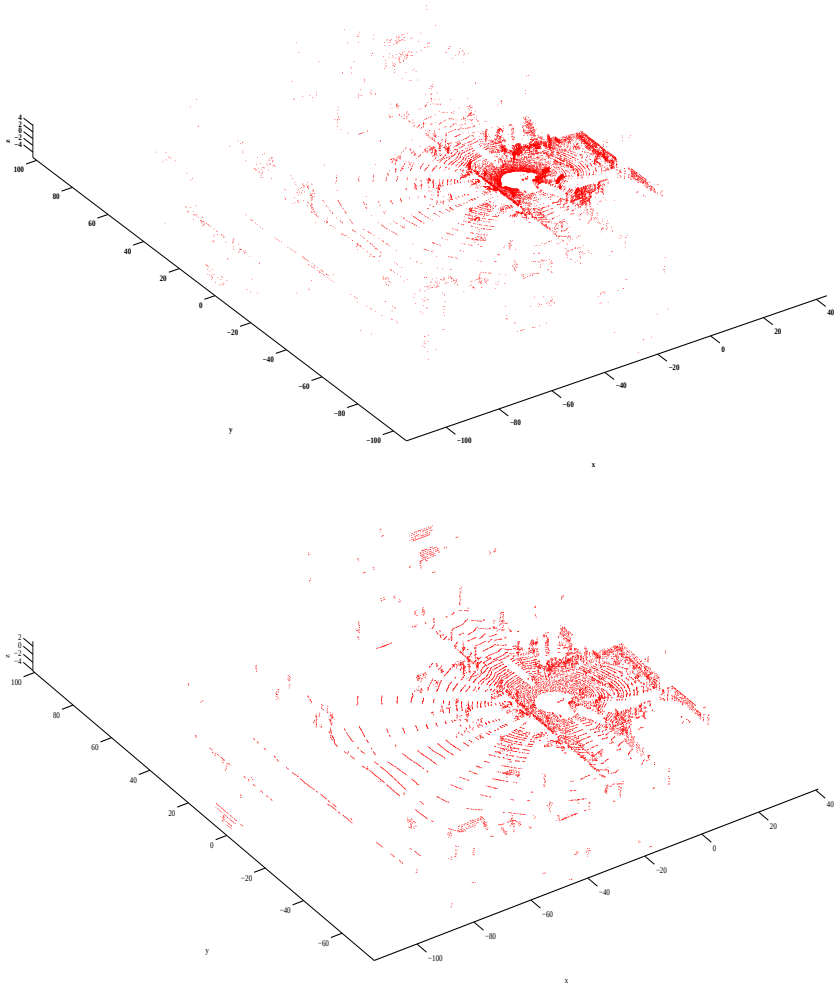


Figure 2.6 : (a)The point cloud [44]. (b)Subsampled with GBS strategy.

2.2.4 Scan Matching Based SLAM

From the SLAM point of view, ICP cannot be directly used to solve SLAM problems, but it is a very important tool. For this purpose, the graph SLAM methods

proposed by Lu and Millos are used in 2D SLAM [31]. In 3D, Nüchter presents linear and closed form solutions to the global *n-scan* registration problem by applying ICP successfully [6]. Also, Ohno proposed a modified ICP method for 3D mapping in real time systems [42]. Pathak introduced a fast 3D mapping method by matching planes extracted from range-sensor point clouds [48] and [47].

Biber introduced *n-scan* matching method for simultaneous matching of multiple scans and application to SLAM with NDT [4]. A recent 3D-SLAM implementation based on 3D NDT is introduced by Magnusson [33]. He uses 3D NDT for scan registration for autonomous mining vehicles and studies on loop closure problem by using NDT [32]. A newsworthy study that combines NDT and Kalman filter is introduced in [22] for 2D SLAM. Based on NDT scan matching, the NDT-EKF carries out rapid and accurate localization by using odometry data and scan matching. Ulas and Temeltas introduced the primitive version of ML-NDT [60] and its application to 3D SLAM problem with a combination of Kalman filter [59]. The details of the ML-NDT and its application to SLAM with Kalman filter is given in Chapter 3.

2.3 Feature Extraction and Feature Based SLAM Methods

While some SLAM methods use raw data with a scan matching algorithm to obtain the rigid body transformations [6], the others use feature extraction algorithms to obtain landmarks from the point clouds [8]. The feature extraction from the LiDAR data is preferred for two main reasons. The first reason is that the number of matched points for scan matching is reduced efficiently and effectively as stated in [62] and [46]. The other important reason is that the features obtained from the LiDAR data can be further processed to obtain landmarks for feature based SLAM [39]. Weingarten and Siegwart proposed an indoor 3D feature extraction method for EKF based 3D SLAM [70]. The method estimates the 6D robot position and model the environment with plane representation. Their method works for the indoor environments and solely estimates the robot pose but not the plane parameters. In plane-feature based representation, the map is very compact, and the environment is modeled by certain geometric characteristics such as infinite plane parameters, plane convex hull points, center of gravity, area, covariance matrix, and the number of inliers as it is achieved in this thesis.

Feature based probabilistic SLAM methods introduced by Thrun have a powerful theoretical background applied to indoor and 2D case successfully in the last decade [51]. Neito et al. used artificial landmarks and natural features like tree for 2D SLAM using particle filters [39]. They also present the results of the algorithm with unknown data association. In their study, both artificial and tree features, which are the center of the trunk, are represented by points. Another method for feature extraction is to convert 3D point cloud into 2D range images and then apply computer vision techniques. Yangming and Olson proposed a general purpose edge-feature detector using computer vision techniques for both 2D and 3D LIDAR data applicable to virtually any environment [75]. Steder et al. has used a similar methodology to propose a point feature extraction method called Normal Aligned Radial Feature (NARF) very recently [53]. The method exploits an interest key point extraction method operating on range images, which explicitly considers the borders of the objects identified by transitions from foreground to background.

Conventional feature extraction in outdoor 3D SLAM is problematic due to the complex nature of the environment, where the randomly distributed data sets representing the irregular objects like bushes and trees disturb the feature extraction. The difficulties of feature extraction in outdoor with respect to indoor is discussed in [76]. The planes are used as rigid objects to increase the performance of SLAM methods, which are mostly preferred as geometric shapes for 3D environment modeling and scan matching. A plane extraction method for scan matching purpose is introduced for indoor and structured environment by Pathak [49]. In addition, Symmetries and Perturbations Map (SPMap) for environment modeling is firstly proposed by Castellanos for 2D case [8], and its extension to EKF based 3D SLAM is presented by Weingarten [70]. Newman et al. combined both camera and LiDAR systems for feature extraction [38]. In their work, they propose a six-degree-of-freedom outdoor navigation system based on vision and laser ranging. Moreover, they explain how their work applied to the topological - metric mapping and pose estimation. With the vision and LiDAR combined framework, they achieved long range SLAM in real time.

Another substantial problem in feature based SLAM is that the vehicle and measurement models are mostly nonlinear. Thus, the extended versions of the filters are used for linearization purposes such as Extended Kalman Filters (EKF) and

Extended Information Filters (EIF) [51]. The Unscented Kalman Filters (UKF) is another filtering technique using unscented transform rather than linearization [24]. This provides better results than EKF if the nonlinear functions are in aggressive fashion [35]. Another important advantage of the UKF is that it does not require any computations of Jacobians, which are the indispensable step for EKF or EIF filters. The filters EKF, EIF, and UKF are known as Gaussian filters since they provide optimum solution under the Gaussian noise. Martinez and Castellanos experimentally validated the usage of UKF for large-scale outdoor environments for point features [35]. For very large scale environments, due to the nature of the EKF and UKF, the state space grows with time and alternative methods like submapping [29] and the usage of sparse filters such as exactly sparse extended information filters (ESEIF) by Matthew [67] and unscented information filters [10] are preferred.

Due to the complexity and unstructured nature of the outdoor environments, it is very difficult to solve data association problem for point features. On the other hand, the appearance based data association using semantic information is recently getting popular since it is more robust and easier than that of point-feature based counterpart. In [38], an appearance based SLAM method using semantic data information for the data association problem is proposed for camera and LiDAR combination. Since the all camera based systems suffers from the illumination problem, McManus show that the successful appearance-based methods traditionally used with cameras can also be applied to only LiDAR [36].

2.4 Discussions

In this section, a history of the SLAM problem is given. The main concepts on scan matching and feature extraction for 6D-SLAM are introduced.

In the first part of this thesis, a scan matching method based on normal distribution transform and its fast and robust version using feature extraction is introduced. Then an application of the scan matching method to SLAM using Kalman filter is introduced.

In the second part of this thesis, two main contributions in feature based SLAM are proposed. Firstly, a reliable landmark detection algorithm based on plane extraction from LiDAR data is introduced. Secondly, these landmarks are used in feature based

3D outdoor SLAM methods, which are based on Local Filters such as EKF, UKF and their variants. In this representation, the SLAM posterior composed of the 6D robot pose and the 4D plane parameters, which are the plane normal (3D) and its minimum distance to origin (1D). The main difficulty with this kind of representation is to solve data association problem since the planes are considered as infinite in the state vector. Thus, the conventional data association algorithms based on statistical methods cannot be used. Instead, the other plane features are used to determine if they belong to the same surface. After the measurement update step, all of the landmark properties are also updated based on the estimated robot pose. Then the associated landmarks are merged to increase the consistency.

3. SCAN MATCHING METHODS AND THEIR APPLICATION TO SLAM

In this chapter, the scan matching methods and their application to SLAM problem is explained. Firstly, Normal Distribution Transform (NDT) based scan matching method proposed by Peter is introduced [3]. Secondly, an improved version of NDT using multi-layer structure, ML-NDT, is elucidated, which is presented by Ulas and Temeltas [59]. Thirdly, further improvements on ML-NDT results in a robust and fast scan matching method called Feature based ML-NDT (FbML-NDT), which uses planar features [62], Fourthly, performance evaluations and sampling strategies are discussed for scan matching in [61]. Finally, the application of the ML-NDT with scan matching Kalman filter to 3D SLAM problem is presented [60].

3.1 Normal Distribution Transform

Normal Distribution Transform (NDT) is described as a set of local probability density functions (pdfs); therefore, a point cloud can be represented in a very compact form. The first step in NDT is to divide the point cloud in to regular cells. Then the mean and covariance values are calculated and stored for each cell. Thus, the point cloud is represented as a sum of normal distributions. If the first and second moments of the discretized reference scan is computed by a *D*-dimensional normal random process, then the probability density of a chosen point, *x*, is computed as

$$p(x) = \frac{1}{(2\pi)^{D/2}\sqrt{|C|}} e^{-\frac{(x-\mu)^T C^{-1}(x-\mu)}{2}} \quad \textbf{(3.1)}$$

where μ and C represent the mean vector and covariance matrix of the reference scan points within a cell including the point *x*. Each pdf can be considered as an approximation of the local surface describing the position of the surface with its orientation and smoothness. The surface orientation and smoothness is determined by the eigenvectors and eigenvalues of the covariance matrix. Depending on the eigenvalues and eigenvectors, the distribution can be a point, line, ellipsoid, and circle shaped in 2D. In 3D, if two eigenvalues are equal and the one eigenvalue is too

small with respect to others, the point cloud distribution denotes a plane. With the similar idea, it is possible to represent other important geometric surfaces like sphere and 3D ellipsoid by normal distribution transform [32].

The mean (μ) and covariance (C) of each cell is defined as follows;

$$\mu_i = \frac{1}{N}\sum_{j=1}^{N} B_{ij} \tag{3.2}$$

$$C_i = \frac{1}{N-1}\sum_{j=1}^{N} (B_{ij} - \mu_i)(B_{ij} - \mu_i)^T \tag{3.3}$$

where i represents the cell index, and j represents point index in each cell. N denotes the number of points in a cell, and $B_{i,j}$ collects all 3D point positions contained in the cell (box). Thus, each cell has the pdfs describing the point cloud in a piecewise smooth representation with continuous derivatives. In Figure 3.1, one can see the normal distributions for a real dataset in 2D. The point cloud has 361 points taken from the experimental Navlab SLAMMOT dataset [68]. In this illustration, the point cloud is divided into 16 equal-sized cells (boxes), and the mean and covariance values of each cell are computed if a cell has more than three points. Finally, the 2D normal distribution function is computed without the normalization constant for all points based on (3.1), and the brighter points indicate higher probabilities. It should be noted that this representation also corresponds to the second layer normal distribution transform of ML-NDT.

The vital part of the NDT method is the discretization process of the point clouds. This discretization artifacts coming from the subdivision process leads to discontinuities in the surface representation and can be problematic in some cases. There are two possible reasons that cause discretization errors. The first one is the selection of the cell size, which describes the resolution. The lower the cell size, the smaller the modeling error is obtained. Therefore, the small sizes should be selected to minimize this kind of discretization or modeling errors. An example of this type of error is seen in the cell centered in (300,-200) of Figure 3.1. The model in this cell contains considerable errors since it does not represent the actual distribution. If the cell sizes are reduces, this error is highly eliminated. Another discretization error occurs at the borders of the cells. Therefore, it is possible to encounter that the pdf for a cell might be non-zero even at points outside the borders of cells as seen in the

cell centered in (100,200) in Figure 3.1. In the conventional 2D NDT implementation, this type of discretization effect is minimized by using four overlapping 2D cell grids with bilinear interpolation [3]. A similar approach is also used by Magnusson in 3D NDT by using eight regular 3D cells with trilinear interpolation [34]. It is shown that this kind of interpolation increases the computations at most eight times.

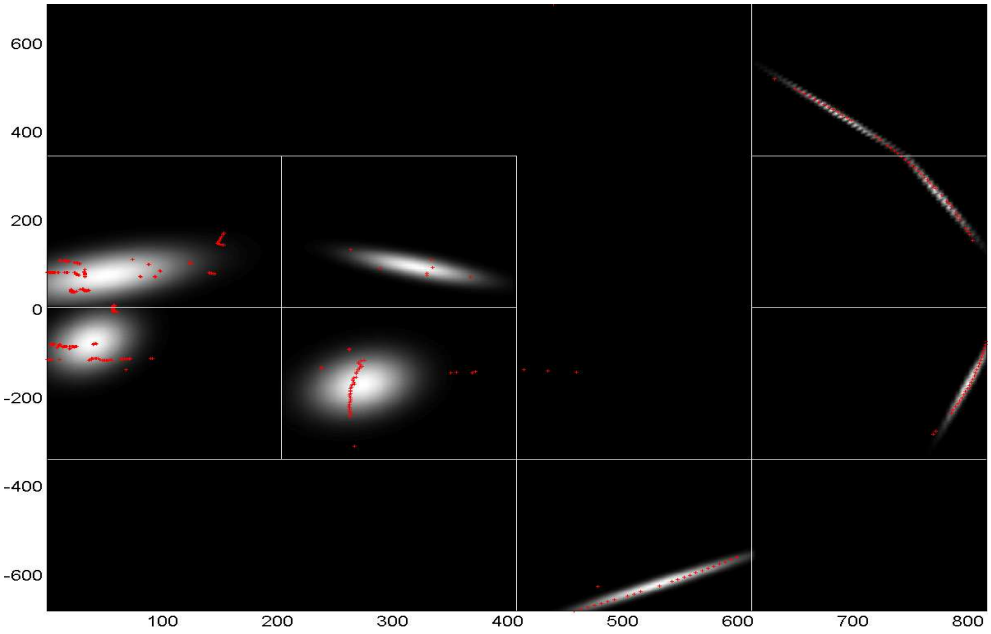


Figure 3.1 : The point cloud (red spots) and its NDT representation.

In the registration process, the algorithm starts with the initial transformation parameters, which are obtained from Odometry or IMU. Then the input scan is transformed with these parameters by using (3.4) and the corresponding cell is found for each point in the input scan. The sum of normal pdf is computed as

$$x' = H(x) = Rx + t \tag{3.4}$$

$$e_i = x_i' - \mu_i \tag{3.5}$$

$$score(p) = \sum_{i=1}^{M} \exp\left(-\frac{e_i^T C_i^{-1} e_i}{2}\right) \tag{3.6}$$

where M is the number of points and i is its index, so x_i' represents the i^{th} point of the transformed input scan. H is the transformation function, which includes both translation, t, and rotation, R. The covariance matrix C_i and the mean μ_i of the corresponding normal distribution to point x_i'.

For more information about the 2D NDT and 3D NDT, we refer the reader to [3], and [32]. An improved version of the NDT, ML-NDT scan matching algorithm, is explained in the next section.

3.2 Multi-Layered Normal Distribution Transform

Choosing the correct cell size is very important for NDT algorithm. For big cells, the convergence time is low; however, the convergence area is large. Therefore, the system mostly results in wrong parameter estimation. However, for the small cell sizes, the convergence time is high, but the accuracy is better for small amount of transformations.

In this paper, a multi-layered structure containing various cell sizes in each layer is proposed. A cell size of a layer is computed from the point cloud (PC) boundary automatically. The point cloud is divided into q^i equal sized cells where $i = 1, 2, .. n$. Here n represents the number of layers, q denotes the division constant in that layer, and i is the layer index. The user initially determine the division and layer numbers like q=8 and n=4. The division number q and number of layers n affect the cell volume in each layer. Therefore, to be able to divide the point cloud in a reasonable cell size in each layer, these values must be reasonable, as well. If the value of q is chosen so small, then the value of n need to be increased, so this increases the data storage and convergence time. In order to keep the algorithm simple, the values of q and n are fixed to 8 and 4 respectively in our experiments. Based on these parameters, the point cloud is divided into 8^i equal-sized cell in each layer. The approach is similar to the octree representation, but the mean vector and covariance matrix of each cell is stored in all layers if there are more than 4 points in a cell. Roughly, the idea is based on the human behavior. First, two objects are compared by looking at the big picture to speed up the perception process and the details are investigated with a fine-tuning by decreasing the searching window. If the PC distribution is similar along the x, y, and z axes, then the PC is divided into fixed cell sizes by dividing each axis into 8^i easily. However, especially in outdoor environments, since there is not much data distributed along the z-axis, the data collected from measurement system is restricted along this axis. Therefore, the PC cannot be divided into 8^i along the z-axis. Assume that a point cloud is distributed along the x and y axes from -50 m to 50 m and along the z-axis from 0 to 20 m. The

enclosed volume of the PC is computed as 100*100*20=200000 m^3. In the second layer (*i=2*) the point cloud must be divided into 8^2 = 64 cells where we want all the cells to have the same volume. Namely, the volume of one cell is computed as 200000/64 = 3125 m^3. Then the one edge of a cube is obtained as $k=3125^{1/3}$ = 14.62m. Thus, the point cloud is divided into *ceil*(100/14.62) = 7 parts along the *x* and *y* axis, and *ceil*(20/14.62)=2 part along the *z* axis. As a result, the point cloud is totally divided into 7*7*2=98 equal-volume cells instead of 64.

The disadvantage part of the algorithm may seem as the data storage; however, the number of cell in the first three layers ($8+8^2+8^3$ =584) is almost ten percent of the only fourth layer, which has 8^4= 4096 cells. If the point cloud contains large amount of data points, the division constant *q* can be made to 10, and the layer number can be increased to five to increase the resolution. We observed that four layers are sufficient in practice. However, to be able to decide which layer to start with for a given initial parameters, we performed an experiment and a guideline is proposed Chapter 5. This guideline recommends that the registration process should starts from the second layer if there is no initial data. If there is odometry data, but it is not in a good quality, then the algorithm should start from the third layer. If the odometry data provides a good initial estimation on the transformation parameters then the algorithm should be started with the forth (or the last layer). Thus, the algorithm becomes more fast and accurate, and it gains “long-range” measurement ability.

The ability of the “long-range scan matching” means that the method can estimate large amount of rotations and translations with respect to traditional methods. While the conventional NDT can estimate small parameter changes, the ML-NDT can estimate large amount of changes. There are two reasons behind this, which are the multi-layered structure and the proposed score function. Choosing the large cell sizes result in better convergence speed but less accuracy, on the other hand using the small cell sizes make the algorithm more accurate but slower. The multi-layered structure combines the advantage of both sides. An important question arises, what is the criterion of switching to the next layer having higher resolutions? In our experiments, we run the algorithm three times in each level, and then go to the next layers except the last layer. In the last layer, the algorithm is run until the convergence occurs or the number of iteration limit is reached. On the other hand, it would be nice to propose a sophisticated switching method based on the behavior

(derivative) of the score function. However, in the experiments it is observed that the score function variation is not always monotonically decreasing. Therefore, it is not plausible to propose a switching criterion based on the score function; therefore, for the sake of simplicity each layer runs three times except the last layer. Since this switching mechanism does not depend on the score function, the information flow between the layers is isolated. The only information shared by the layers is the point cloud, and there is no direct relation among the layers. In the last layer where it has the highest resolution, the algorithm runs until the convergence criteria are met. The convergence criteria depend on two factors that are the norm of the amount of update and the number of iteration. Another reason is why ML-NDT outperforms the NDT is the score functions used for optimization. In traditional NDT, the score function given by (3.1) is optimized by increasing the probability of input scan points matched on the reference model, on the other hand, the ML-NDT score function given by (3.6) is optimized by decreasing the distance of input scan points matched on the reference model. As a result, the multi-layered structure with the optimization of Mahalanobis distance score function outperforms the conventional NDT. The performance of the long-range measurement ability of the ML-NDT over NDT is discussed in Chapter 5.

To illustrate the multi-layered structure, the first three layer cell boundaries and their normal distribution transform are depicted for 2D point cloud in Figure 3.2. In each layer, the point cloud is subdivided into 4^i equal parts, where i is layer index with $i=1, 2, 3$. Each layer is drawn with a different line type and color.

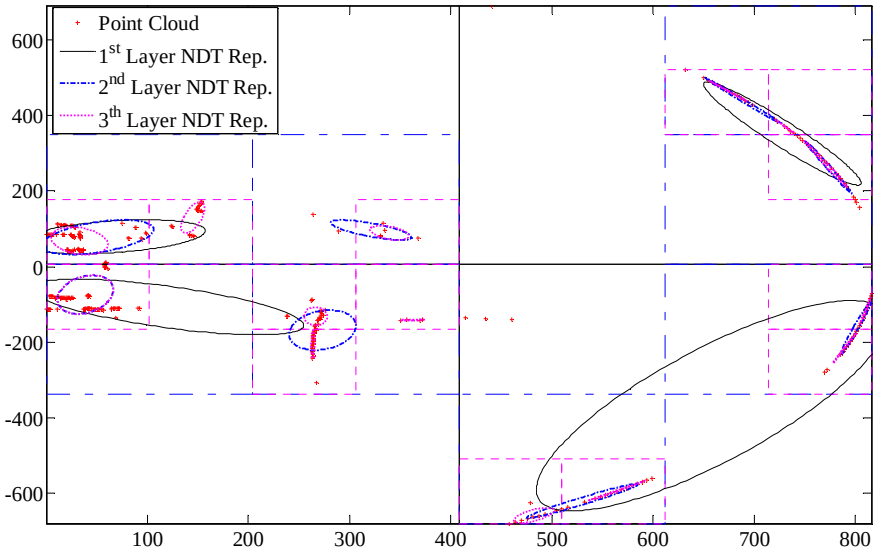


Figure 3.2 : Multi-Layered NDT representations.

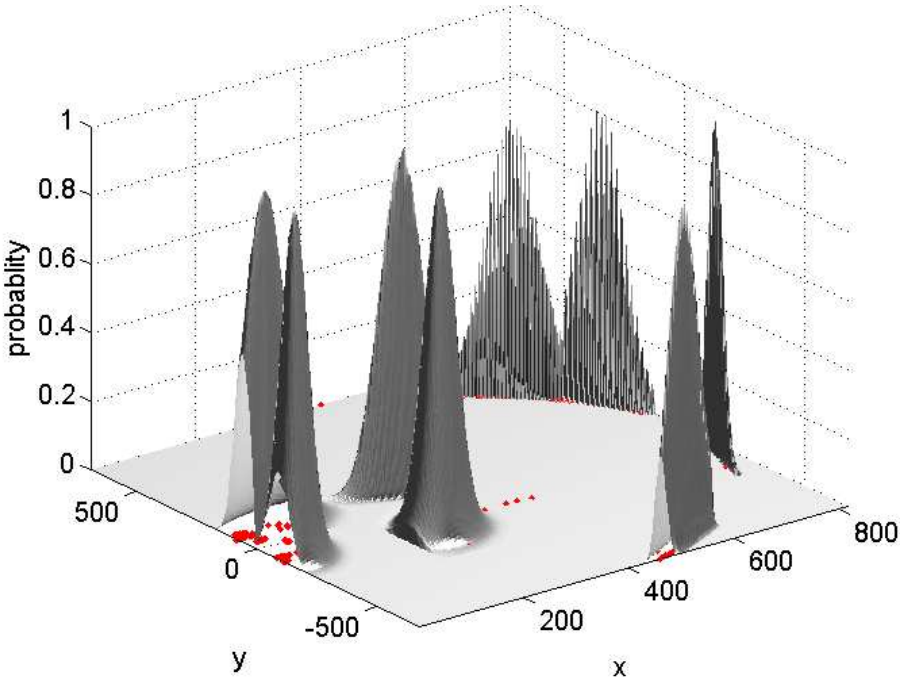


Figure 3.3 : 2nd layer surface representation of the probability function (3.1).

While the first layer has the largest cells, the third layer has the smallest cells. Since this representation in 2D, there are 4 cells in the first layer, $4^2=16$ cell in the second layer, and $4^3=64$ cell in the third layer. Figure 3.2, the cell boundaries are drawn only for occupied cells and they are only for illustrations.

A surface representation of the point cloud shown in Figure 3.1 is given in Figure 3.3. Here, the second layer normal distribution, which includes *16* cells, is computed for illustration. The ML-NDT algorithm and its steps, which are splitting process, generative process, and registration process, are given in Table 3.1. Before the splitting process, a layer number must be defined by the operator. Then the point cloud limits are calculated, and left-down and up-right corners of each cell in all layers are computed. In the splitting process for all layers, the point cloud is distributed to equal sized cells. In generative process, for all layers and each cell, covariance matrices and mean vectors are calculated and stored to *B* cell array if a cell contains more than *4* points. Finally, in registration process, the algorithm is initialized with a starting layer *s* and the conventional NDT is applied to this layer. The algorithm runs three times in this layer, and it is switched to next layer. The same procedure is applied to each intermediate layers, and finally the algorithm returns relative transformation parameters ξ when the convergence criteria are met.

Table 3.1 : Multi-Layered Normal Distribution Transform (ML-NDT) Algorithm.

Algorithm $(\xi) = \textbf{ML - NDT}(R, S, q, n, s)$

Inputs:

R: Reference Scan
S: Input Scan
q: Division constant
n: Number of layers
s: Starting layer

Outputs:

ξ: Relative transformation parameters (translations and rotations)

Splitting Process

1: for all layers l_i *(i=1,2,..n)* do
2: Divide the point cloud into q^i cells.
3: for all $r_i \epsilon R$ do (r_i is the i^{th} point)
4: Find the cell c_j that includes r_i, then store r_i in c_j
5: end for
6: end for

Generative Process

1: for all layers l_i *(i=1,2,..n)* do
2: for all cells $c_j \epsilon C$ do
3: if number of points in $c_j > 4$
4: Calculate the covariance and mean values of each cell c_j and store to *B*.
5: end if
6: end for
7: end for

Registration Process

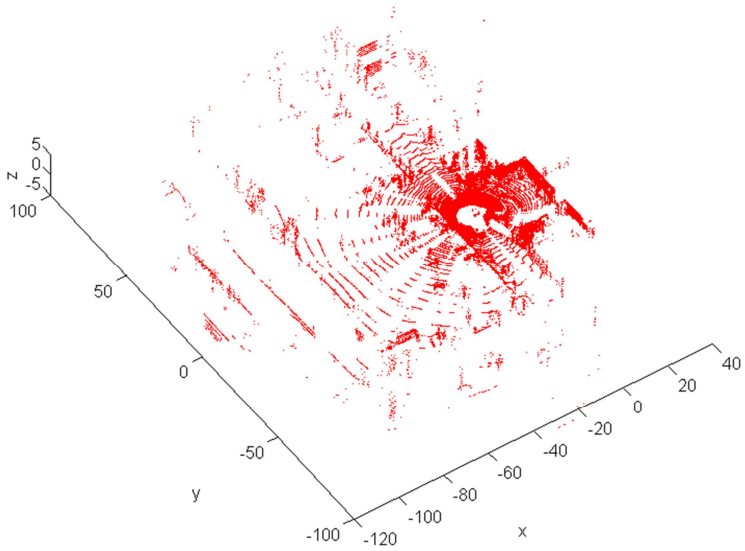
1: Start with the layer *s*
2: Wait for the convergence
3: for each point in $s_i \epsilon S$ do (s_i is the i^{th} point)
4: find the corresponding cell in the reference model *B*
5: Update ξ by following equations (6) through (17).
6: Switch to the next layer in every three iterations.
7: end for
8: return ξ

In the algorithm, *S* denotes the input scan points used in the matching. Therefore, *S* represents the subsampled of points from the input scan. In practice, up to 10% of the input scan is sufficient for successful convergence. The problem of choosing the

subsamples is discussed in and the effect of sampling strategies on the performance is given in Chapter 5.

To show 3D ML-NDT generative process, an experimentally obtained point cloud from Ford Dataset [44] is used. The point cloud including 74951 points and its fifth layer NDT representation is given in Figure 3.4 (a) and Figure 3.4 (b), respectively. That is, the point cloud is divided into 100.000 cells and occupied cells are shown for better illustration.

(a)



(b)

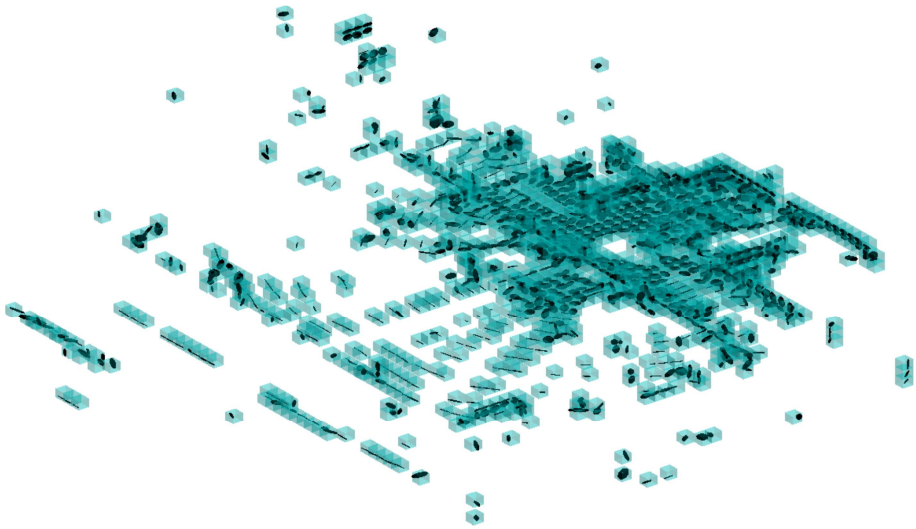


Figure 3.4 : (a)A 3D scan. (b)Its fifth layer NDT representation.

There are important differences in our formulations from the original NDT since we use Mahalanobis distance score function instead of a probabilistic one. Thus, the formulations become easier; hence, it takes less computation time. In addition, the

algorithm can estimate long-range transformations. Derivations of equations for ML-NDT are explained as follows.

The equation given by (3.4) is rewritten in (3.7) for convenience. Here R represents the rotation matrix containing the Euler angles (α, β, γ) in (3.8). The translation vector t is given by (3.9).

$$x' = Rx + t \tag{3.7}$$

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\alpha & -s\alpha \\ 0 & s\alpha & c\alpha \end{bmatrix} \begin{bmatrix} c\beta & 0 & s\beta \\ 0 & 1 & 0 \\ -s\beta & 0 & 0 \end{bmatrix} \begin{bmatrix} c\gamma & -s\gamma & 0 \\ s\gamma & c\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.8}$$

$$t = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{3.9}$$

where $\boldsymbol{x} = (x, y, z)$ is an input scan point and x' is the transformed point based on the initial parameters provided by the odometry. Initial values of rotations and translations are accepted as zero if there is no odometry or IMU sensor information. In fact, all scan matching algorithms, ICP, NDT, and ML-NDT requires these initial values. However, we observed that in some cases, the ML-NDT converges to the correct transformation parameters without any initial parameters.

Let e be the error vector between transformed point and the mean of the corresponding cell. The Mahalanobis distance score function based on this error e and the covariance C is described by (3.11).

$$e = x' - \mu_i \tag{3.10}$$

$$Score = \sum_{k=1}^{M} e^T C^{-1} e \tag{3.11}$$

Another important issue is to get rid of singularities on the covariance matrix C. The computation of the score function (3.11) requires the inverse of C. The matrix C may easily become singular in case the point distribution in a cell is perfectly collinear or coplanar. A solution is proposed by Magnusson, who slightly inflated the covariance matrix (C) whenever it is found to be nearly singular [32]. If the largest eigenvalue of C is more than 100 times greater than l_j where $j=1,2$ represents the other eigenvalue

indices, then the eigenvalues are replaced with $l'_j = l_3/100$. The new covariance matrix is replaced with $C' = V\Lambda'V$, with V containing the eigenvectors of C and $\Lambda' = diag(l'_1, l'_2, l_3)$.

Rigid body transformation parameters searched for are collected in a vector as given in (3.12) and estimated iteratively by using Newton and Levenberg–Marquardt optimization methods, respectively.

$$\xi = [t_x\, t_y\, t_z\ \ \alpha\, \beta\, \gamma]^T \tag{3.12}$$

The rest of derivations of the formulations depend on the optimization method. For this reason, this section is subdivided into two parts in order to explain both methods. In the next two subsections, the optimization methods used by the algorithm are given, which are Newton and Levenberg-Marquardt iterative optimization methods.

3.2.1 Optimization with Newton Method

To be able to minimize the score function, the gradient vector g_k, and the Hessian matrix H_k of the summand of the score function s_k is written as the following equations.

$$s_k = e^T C^{-1} e \tag{3.13}$$

$$g_k = \frac{\partial s_k}{\partial \xi} = \frac{\partial s_k}{\partial e}\frac{\partial e}{\partial \xi} = e^T C^{-1}\frac{\partial e}{\partial \xi} \tag{3.14}$$

$$H_k = \frac{\partial^2 s_k}{\partial \xi_i \partial \xi_j} = \left(\frac{\partial e}{\partial \xi_j}\right)^T C^{-1}\frac{\partial e}{\partial \xi_i} + e^T C^{-1}\frac{\partial^2 e}{\partial \xi_i \partial \xi_j} \tag{3.15}$$

For derivations of partial deriatives of $\frac{\partial e}{\partial \xi_i}$ and $\frac{\partial^2 e}{\partial \xi_i \partial \xi_j}$, one can look at [32]. The transformation parameters ξ_t are updated iteratively as

$$\xi_t = \xi_{t-1} - H^{-1} g \tag{3.16}$$

where g and H are computed by (3.17) and (3.18), respectively.

$$g = \sum_{k=1}^{M} g_k \tag{3.17}$$

$$H = \sum_{k=1}^{M} H_k \tag{3.18}$$

In some cases, *H* might be very close to singularity, or it might be negative definite. However, it has to be positive definite and non-singular for a successful optimization. If *H* is positive definite, the score function will decrease in the direction of $\Delta \xi = \xi_t - \xi_{t-1}$. If this is not the case, *H* is replaced by $H = H + \gamma I$, with γ chosen such that *H* is safely positive definite. One can see [3] for more details.

3.2.2 Optimization with Levenberg-Marquardt Method

Levenberg-Marquardt (LM) method combines the advantages of the steepest decent and Gauss-Newton methods. Therefore, it might provide faster convergence time. The method is integrated to ML-NDT in this section.

The score values of each point is represented by s_k , and the score vector *S* consists of this score values;

$$S = [s_1\, s_2 \ldots\ldots s_k \ldots . s_M]^T \tag{3.19}$$

Let J be the Jacobian matrix with *Mx6* dimensions,

$$J_{kj} = \frac{\partial s_k}{\partial \xi_j} = \frac{\partial s_k}{\partial e}\frac{\partial e}{\partial \xi_j} = e^T C^{-1}\frac{\partial e}{\partial \xi_j} \tag{3.20}$$

In Gauss-Newton method, Hessian matrix is approximated by $J^T J$, and step size λ is used in steepest descent method. The LM algorithm combines these two methods into one form. As a result, the time update is accomplished as

$$\xi_t = \xi_{t-1} - (J^T J + \lambda I)^{-1} J^T S \tag{3.21}$$

where λ is initially chosen as 10^{-3} times the average of the diagonal elements of approximated Hessian matrix. Then λ is updated depending on the variations of the score function. If the error becomes smaller, λ is divided by a constant (typically, 10), else it is multiplied by this constant.

In the next section, an improved version of the ML-NDT for fast and robust scan matching is presented.

3.3 A Fast and Robust Feature Based Scan Matching Algorithm

In this section, the details of the feature-based scan matching method, FbML-NDT, are explained. First, the scan matching and the plane-feature extraction methods for 3D SLAM are explained separately. Then using these feature points in the registration process of the scan matching algorithm, a feature based scan matching method is obtained. Finally, the extracted plane segments are merged in a dynamic region-growing algorithm to obtain planar landmarks for feature based SLAM methods. As a result, a combined method that merges the scan matching and feature extraction methods into a single algorithm is obtained. This study is published in [61], which is proposed by Ulas and Temeltas.

3.3.1 Feature Extraction

The steps of the 3D robust feature extraction algorithm are explained in this section. The method starts with point cloud discretization by dividing the point cloud into regular cells and assigning each point into these cells. This step is also known as splitting process. The well-known and simple discretization method is to use fixed cells; however, there are other methods such as tree representations, iterative discretization, and adaptive clustering. Fixed cells are used for the sake of its simplicity; however, using other methods might be interesting. Since main contribution is not in this part, we refer the reader to the literature.

3.3.1.1 Probabilistic Plane Fitting

After the discretization process, planar segments are detected by first finding the inliers of a plane via RANSAC algorithm [16]. Then if the number of inliers in a cell exceeds a threshold value (practically 10), the least mean square plane fitting is applied to these inliers to obtain parametric representation of the plane.

A general plane equation can be written as follow;

$$\beta_a x + \beta_b y + \beta_c z + \beta_d = 0 \qquad \textbf{(3.22)}$$

where β vector represents the infinite plane parameters. While the first three elements of β vector denote to normal vector (n), the distance to origin is represented by the last element of the β vector. The constraint matrix is composed of as follow;

$$A = \left[\Gamma_x \, \Gamma_y \, \Gamma_z \, 1 \right] \tag{3.23}$$

where Γ indicates the columns of the plane data points which is given in x, y, and z axes. Last column of the constraint matrix A is filled by ones, and then singular value decomposition (SVD) of A yields the plane parameters. As a result, the third column of the eigenvector $V \in R^3$ is equal the β vector.

$$A = UEV^T \tag{3.24}$$

$$\beta = V_{j4} \qquad (j=1, 2, 3) \tag{3.25}$$

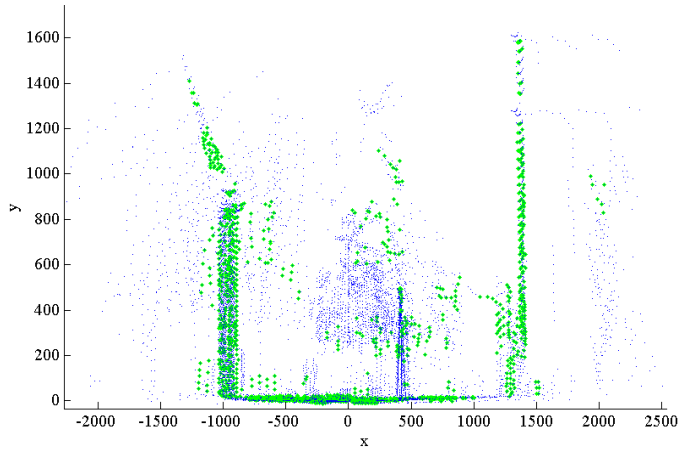


Figure 3.5 : Feature points extracted from the point cloud.

If there are large point sets and the outliers exist, this method becomes defective in terms of computational burden and accuracy. Instead, Principal Component Analysis (PCA) can be used. The covariance matrix of all points belonging to the plane is found. Then SVD is applied to covariance matrix. The last column corresponding to smallest variation axis gives the normal vector of the plane. The parameter β_d is found by putting any plane point into the (3.22). The extracted plane points from the point cloud provided by Hannover dataset are shown in Figure 3.5. The size of the points to be matched is almost 10% of the original point cloud.

3.3.1.2 Projection and Convex Hull Computation

After finding the infinite plane parameters, the convex hull or the minimum rectangular bounding box of the planes is computed. Both convex hull and minimum bounding box computation is based on the projection of the inlier plane points into a 2D space. In literature, the projection methods project the plane points onto x-y axis [26]. However, since the orientation of the plane points in the space might be various, we propose a PCA based projection method. The plane points are projected into their principal components by finding the rotation matrix and translation vector. The projection and convex hull computation method is explained as follows;

The rotation matrix (*R*) and the translation vector (*T*) can be computed as follows;

First the plane points is translated to the first element position by the following subtraction,

$$B = \Gamma - \Gamma_0 \tag{3.26}$$

where Γ_0 is the first point of the point cloud and it is considered as the translation vector (T). The mean vector, $\bar{\mu}$, and covariance matrix, C of the plane points are computed as follows;

$$\bar{\mu} = \frac{1}{N}\sum_{j=1}^{N} B_i^{\,j} \tag{3.27}$$

$$C = \frac{1}{N-1}\sum_{j=1}^{N}(B_i^{\,j} - \bar{\mu})(B_i^{\,j} - \bar{\mu})^T \tag{3.28}$$

The SVD is applied to the covariance matrix, C.

$$C = UEV^T \tag{3.29}$$

The rotation matrix (*R*) becomes the normalized first two columns of the eigenvector matrix, $V = [\vec{v}_1 \ \vec{v}_2 \ \vec{v}_3]$, which are the principal axes of the plane points.

$$R = [\vec{v}_1 \ \vec{v}_2] \tag{3.30}$$

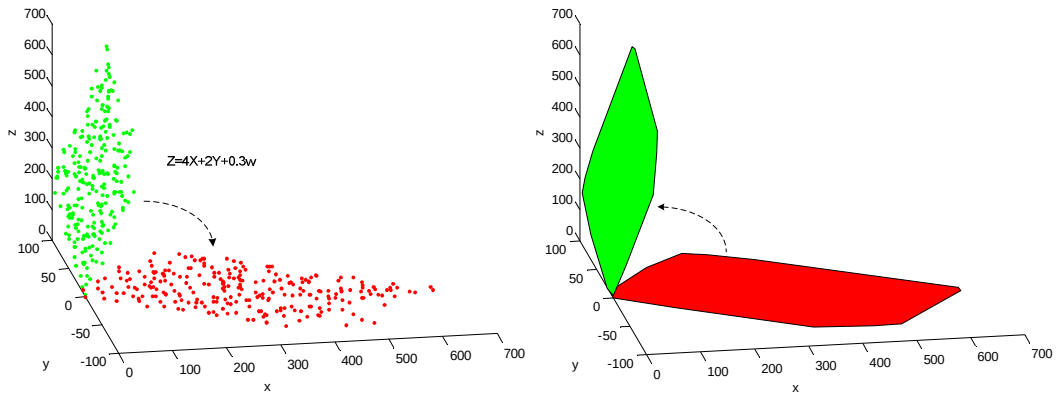
The translated plane points B is projected to the principal axis plane by using this rotation matrix (R). The 3D plane points are now in 2D space and represented as $\Delta_{XY} \in \mathrm{R}^2$.

$$\Delta_{XY} = RB \tag{3.31}$$

Then the convex hull of the projected plane points Δ_{XY} [35] or minimum rectangle-bounding box is found. Then the convex hull points, H_p are translated back to 3D space with the following expression and represented as $\Delta_{XYZ} \in \mathrm{R}^3$.

$$\Delta_{XYZ} = H_p R' + \Gamma_0 \tag{3.32}$$

Figure 3.6 (a) shows the 3D plane points and its translated and projected views and they are transformed back to the 3D space as shown in Figure 3.6 (b).



(a) (b)

Figure 3.6 : Convex hull computation: (a)Plane and its projection to principal axis plane. (b)Convex hull computation in 2D and back-projection to 3D.

To show an application of the plane extraction method in outdoor, Ford dataset [44] is used. The extracted plane segments of the environment are given in Figure 3.7. The camera images of the environment and its point cloud representation that is projected to these camera images are shown in Figure 3.8.

3.3.1.3 Feature Based Multi-Layered Normal Distribution Transform

In this part, a novel scan matching method using ML-NDT algorithm based on extracted features, called FbML-NDT, is introduced. For the reference scan, the regular ML-NDT is applied to each layer, and for the input scan, which is the one to

be aligned to reference scan, the plane inliers are extracted as explained in the previous section. Then for each plane points, the regular registration process is applied as given in Table 3.1.

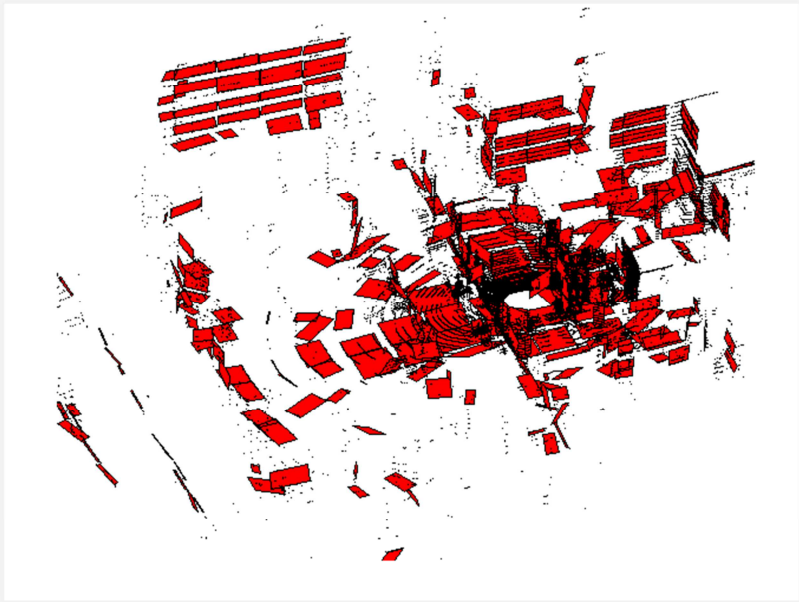


Figure 3.7 : The extraction of the plane segments.

(a)

(b)

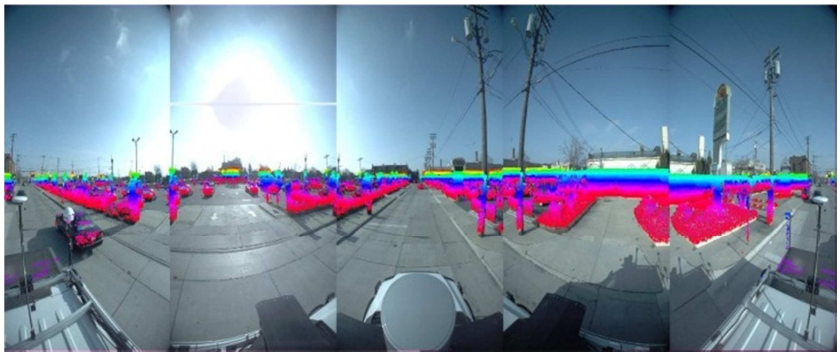


Figure 3.8 : Camera Vision: (a)Ladybug camera vision of the environment. (b)The point cloud projection into the corresponding images.

Table 3.2 : FbML-NDT Algorithm.

Algorithm $(\xi) = \textbf{FbML - NDT}(R, S, q, n, s)$

Inputs:

R: Reference Scan
S: Input Scan
q: Division constant
n: Number of layers
s: Starting layer

Outputs:

ξ : Relative transformation parameters (translations and rotations)

Splitting Process $(C) = \textbf{Split}(R, q, n)$

1: for all layers l_i *(i=1,2,..n)* do
2: Divide the point cloud into q^i cells.
3: for all $r_i \epsilon R$ do (r_i is the i^{th} point)
4: Find the cell c_j that includes r_i, then store r_i in c_j
5: end for
6: end for

Generative Process $(B) = \textbf{GeneratePDF}(C, n)$

1: for all layers l_i *(i=1,2,..n)* do
2: for all cells $c_j \epsilon C$ do
3: if number of points in $c_j > 4$
4: Calculate the covariance and mean values of each cell c_j and store to *B*.
5: end if
6: end for
7: end for

Feature Extraction Process $(F) = \textbf{FeatureExtraction}(S)$

1: $(C) = \textbf{Split}(S, q, n)$
2: Apply RANSAC algorithm and obtain plane inlier points
3: Call this inlier points as *F*

Registration Process $(\xi) = \textbf{Registration}(B, F, s)$

1: Start with the layer *s*
2: Wait for the convergence
3: for each point in $f_i \epsilon F$ do (f_i is the i^{th} point)
4: find the corresponding cell in the reference model *B*
5: Update ξ by following equations (4) through (12).
6: Switch to the next layer in every three iterations.
7: end for
8: return ξ

The feature based ML-NDT (FbML-NDT) algorithm given in Table 3.2 consists of the four phases: the splitting process, the generative process, feature extraction, and the registration process. In the splitting process the reference scan is divided into fixed cells, and in the generative process the mean vector and the covariance matrices of each cell is computed as in ML-NDT. In feature extraction process, the input scan is also discretized and the plane candidates are detected via RANSAC algorithm [16].

If the number of inliers in a plane is greater than a threshold value, these plane inlier points are considered as feature points (F) for registration process. In the registration process, the extracted feature points are used in the optimization step of the ML-NDT algorithm; thus, the number of matching points is reduced effectively and efficiently. The rigid body transformation parameters ξ are obtained from the proposed algorithm.

The feature extraction for scan matching can be considered as a special case of subsampling method. This is true in terms of point reduction in the registration process of the scan matching methods. However, since the feature extraction based scan matching utilizing the geometric structures, which are planes in this study, they are more robust than the conventional methods even if they are sampled with GBS strategy. Another issue is that GBS method samples equal number of points from all cells and does not take the importance of a cell into account. In Chapter 5, the performance analysis of the feature based scan matching and the effect of sampling strategies is given.

3.4 Application of Scan Matching to SLAM Problem

In this section, an application of a scan matching method to solve SLAM problem is explained. The proposed Multi-Layered NDT scan matching algorithm is used to match two consecutive scans to find relative rigid body transformation parameters, which are then used to estimate robot path and the dense map of the environment. One has to note that only the last robot pose is estimated and updated as in online SLAM and the state vector size is constant and equals to the number of robot poses. If the initial pose (position and orientation) of a mobile robot is known, one can find the next pose of the mobile robot by using the relative transformation parameters obtained by the ML-NDT scan matching algorithm. This part does not use any

sophisticated filtering methods, such as Kalman filter, particle filters, or information filters. Therefore, the path obtained by the proposed algorithm here contains cumulative error.

The combination of ML-NDT with Kalman filter provides better results for estimating the robot pose. The proposed method is explained as follows;

Let the initial pose of the robot is given as

$$x_0 = \begin{bmatrix} x\ y\ z\ r_x\ r_y\ r_z \end{bmatrix} \tag{3.33}$$

where *x*, *y*, *z* are the position of the robot, and $r_x\ r_y\ r_z$ are the Euler angles, which represent the orientation of the mobile robot in the global frame. If the relative transformation, combination of rotation (*R)* and translation (*T)*, is known, the next pose of the vehicle in global frame can be found as follows;

$$x_{t+1} = Trans3D(x_t, \xi) = Rx_t + T \tag{3.34}$$

This is applied recursively to find the consecutive poses of the mobile robot. Then t[th] scan $scan_t^L$ in local frame (*L*) is projected to world (*W*) frame as $scan_t^W$ if the global robot pose x_t is known.

$$scan_t^W = Trans3D(scan_t^L, x_t) \tag{3.35}$$

The dense map can be obtained by putting together the all projected scans into global frame.

$$Map = \begin{bmatrix} scan_0^W \\ scan_1^W \\ . \\ . \\ scan_N^W \end{bmatrix} \tag{3.36}$$

3.4.1 Application of Scan Matching to SLAM Problem in 2D

As a vehicle, Ackermann wheel model is used and the observation is based on the relative transformation parameters obtained from scan matching. The kinematic motion model and observation model of the mobile robot are defined as follows;

Kinematic Model The vehicle motion model shown in is given in (3.37).

$$\begin{aligned} x_{t+1} &= x_t + V_k dt\cos(\theta_{v_k} + \gamma_k) \\ y_{t+1} &= y_t + V_k dt\sin(\theta_{v_k} + \gamma_k) \\ \theta_{v_{k+1}} &= \theta_{v_k} + \frac{V_k dt}{L}\sin(\gamma_k) \end{aligned} \tag{3.37}$$

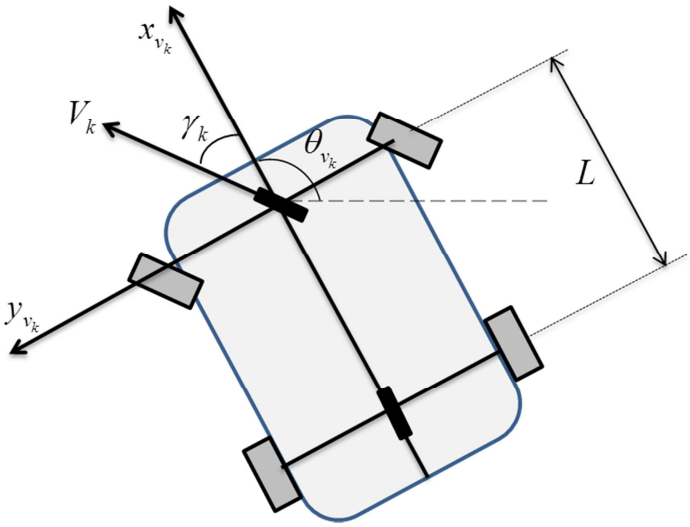


Figure 3.9 : Discrete Time Vehicle Motion Model.

Jacobian matrix of the motion model and control models, which are shown by G_v and G_u, respectively, is found from the kinematic model (3.37) as

$$G_v = \begin{pmatrix} 1 & 0 & -V_k dt\sin(\theta_{v_k} + \gamma_k) \\ 0 & 1 & V_k dt\cos(\theta_{v_k} + \gamma_k) \\ 0 & 0 & -1 \end{pmatrix} \tag{3.38}$$

$$G_u = \begin{pmatrix} dt\cos(\theta_{v_k} + \gamma_k) & -V_k dt\sin(\theta_{v_k} + \gamma_k) \\ dt\sin(\theta_{v_k} + \gamma_k) & V_k dt\cos(\theta_{v_k} + \gamma_k) \\ \frac{dt}{L}\sin(\gamma_k) & \frac{V_k dt}{L}\cos(\gamma_k) \end{pmatrix} \tag{3.39}$$

Measurement unit provides the robot relative transformation parameters from scan matching as given (3.40). Therefore, we can find the observation model ξ as

$$\xi = [t_x \quad t_y \quad \gamma \ ]^T \tag{3.40}$$

$$x_{t+1} = x_t + \begin{pmatrix} cos(\theta_t) & -sin(\theta_t) \\ sin(\theta_t) & cos(\theta_t) \end{pmatrix} \begin{pmatrix} t_x \\ t_y \end{pmatrix} \tag{3.41}$$

$$\theta_{t+1} = \theta_t + \gamma \tag{3.42}$$

where t_x and t_y are the relative translations, and γ is the relative orientation. They are extracted from (3.41) and (3.42) as

$$\begin{pmatrix} t_x \\ t_y \end{pmatrix} = \begin{pmatrix} cos(\theta_t) & -sin(\theta_t) \\ sin(\theta_t) & cos(\theta_t) \end{pmatrix}^{-1} (x_{t+1} - x_t) = \begin{pmatrix} cos(\theta_t) & sin(\theta_t) \\ -sin(\theta_t) & cos(\theta_t) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \tag{3.43}$$

$$\begin{pmatrix} t_x \\ t_y \end{pmatrix} = \begin{pmatrix} \Delta x cos(\theta_t) + \Delta y sin(\theta_t) \\ -\Delta x sin(\theta_t) + \Delta y cos(\theta_t) \end{pmatrix} \tag{3.44}$$

$$\gamma = \theta_{t+1} - \theta_t. \tag{3.45}$$

Observation Model The observation model is given as

$$\xi = \begin{pmatrix} t_x \\ t_y \\ \gamma \end{pmatrix} = \begin{pmatrix} \Delta x cos(\theta_t) + \Delta y sin(\theta_t) \\ -\Delta x sin(\theta_t) + \Delta y cos(\theta_t) \\ \theta_{t+1} - \theta_t \end{pmatrix} \tag{3.46}$$

Jacobian matrix of the observation model is found from the observation model.

$$H = \begin{pmatrix} -cos(\theta_t) & -sin(\theta_t) & -\Delta x sin(\theta_t) + \Delta y cos(\theta_t) \\ sin(\theta_t) & -cos(\theta_t) & -\Delta x cos(\theta_t) + \Delta y sin(\theta_t) \\ 0 & 0 & -1 \end{pmatrix} \tag{3.47}$$

Motion Update The state and covariance updates are given as

$$\begin{aligned} x_{t+1} &= x_t + V_k dt cos(\theta_{v_k} + \gamma_k) \\ y_{t+1} &= y_t + V_k dt sin(\theta_{v_k} + \gamma_k) \\ \theta_{v_{k+1}} &= \theta_{v_k} + \frac{V_k dt}{L} sin(\gamma_k) \end{aligned} \tag{3.48}$$

$$\Sigma_{t+1} = G_v \Sigma_t G_v^T + G_u Q G_u^T \tag{3.49}$$

where, Q is the control noise and Σ_t is the covariance matrix at time t.

Measurement Update Update equations requires the computations of Kalman gain, and it is found by

$$K = \Sigma_{t+1} H^T (H \Sigma_{t+1} H^T + R)^{-1} \quad \textbf{(3.50)}$$

where R is the measurement noise obtained from the ML-NDT algorithm by taking the inverse of the hessian matrix given in (3.18), $R = H^{-1}$, and the proof is given in [4]. The reader should be careful with the notation of Hessian matrix and the Jacobian of the observation model represented by the same symbol. The measurement update equations of the state vector and covariance matrix is given by

$$\hat{X}_{t+1} = X_{t+1} + K(z_m - z) \quad \textbf{(3.51)}$$

$$\hat{\Sigma}_{t+1} = \left(I - KH\right)\Sigma_{t+1} \quad \textbf{(3.52)}$$

3.4.2 Application of Scan Matching to SLAM Problem in 3D

The idea is principally is the same as the 2D case and its extension to 3D. Here only the differences and the assumptions are discussed for 3D case.

To be able to implement a Kalman filter based SLAM, one has to know the state space representation of the controlled system and observation models as well as the noise levels with their covariance values. The datasets might provide odometry data in three dimensions as x, y, and θ. This odometry data, which provides the robot position in every time stamps, is used to derive the control signal. Hence, the kinematic model can be considered as linear in time update phase of the KF, so there is no need to compute Jacobian matrices because they are identity matrices. Again, the datasets mostly does not provide the covariance matrix of the noise of the control signal, and it is assumed that each diagonal element of the error covariance matrix of the odometry is assumed as a fixed value and other entries are zero. Although the measurement method, ML-NDT, provides relative homogenous transformation between two successive scans, this roto-translation can be used as positioning technique. Since scan registration method provides the next position of the vehicle based on its previous position, the observation model can be considered as a linear model. The control signal is obtained from the odometry and used in KF time update step as shown in Table 3.3.

Table 3.3 : Kalman Filter time update and measurement update steps.

Time Update

$\hat{x}(k+1|k) = \hat{x}(k|k) + u(k)$

$P(k+1|k) = P(k|k) + Q$

Control signal obtained from odometry:

$u(k) = x_o(k) - x_o(k-1)$

Measurement prediction:

$\hat{z}(k+1|k) = x_o(k)$

Measurement Update

$\hat{x}(k+1 \mid k+1) = \hat{x}(k+1|k) + K\varepsilon$

$P(k+1|k+1) = (I - K)P(k+1|k)$

Innovation (ε), Innovation Matrix (S), and Kalman Gain (K):

$\varepsilon = z(k+1) - \hat{z}(k+1|k)$

$S = P\hat{x} + R \qquad K = P\hat{x}S^{-1}$

$\hat{P}(k+1|k) = \hat{P}(k) + Q$

3.5 Discussions

In this chapter, first, a novel scan matching method, ML-NDT, based on Normal Distribution Transform is introduced. This algorithm uses multi-layered structure that automatically calculates the cell size from the point cloud boundaries instead of assigning a fixed cell size. Then an extension of ML-NDT using planar features for a fast and robust scan registration is proposed. Since this method uses certain geometric shapes in registration, the performance of the algorithm significantly increases. Finally, the application of the proposed scan matching methods to 6D SLAM problem is explained. The experimental results of both scan matching and scan matching based SLAM is given and discussed in the Chapter 5.

4. A PLANE-FEATURE EXTRACTION METHOD AND ITS APPLICATION TO 6D SLAM

This chapter is divided into two sections: First, the planar feature extraction method is introduced. Second, the feature based SLAM in terms of planar features is proposed. The work comprising of the feature extraction methods and its application to SLAM with local filters is also published in [66] and [64], respectively. In addition, the method is combined with Cubature Kalman Filters (CKF) and semantic perception for outdoor environments and published in [63]. Moreover, a novel filter, Randomized Sigma Point Kalman Filter (RSPKF) for feature based 6D-SLAM is published by Ulas and Temeltas [65]. Finally, sparse extended information filters are adapted to planar-feature extraction algorithm for large-scale SLAM. The details of the proposed methods are given in this section.

4.1 Plane-Feature Extraction Method

In this section, the steps of the 3D landmark extraction algorithm are explained. A fundamental part of the landmark extraction algorithm is presented in the Chapter 3 and Section 3.1.1 for fast and robust scan matching. Here, the feature extraction algorithm is improved for planar feature based SLAM method. After the projection and convex hull computation step, the plane segments are merged to obtain large plane structures. The merging step is given as follows.

Merging Step After finding the convex hull points of the planes, the plane segments are merged if they pass the three merging tests. These tests are given as follows,

- *Orientation test* $C_o = |\, cos^{-1}\{\mathbf{n}_i^{\mathrm{T}}.\mathbf{n}_j\}| < \mathrm{t_O}$
- *Translation test* $C_t = |\, \mathbf{n}_i^{\mathrm{T}}(\mathbf{c}_i - \mathbf{c}_j)| < \mathrm{t_T}$
- ***Closeness test*** $C_p = (\mathbf{c}_i - \mathbf{c}_j)^T \Sigma_p^{-1} (\mathbf{c}_i - \mathbf{c}_j) < \mathrm{t_P}$

where c_i is the center of the gravity of a plane, Σ_p is the pooled covariance matrix as defined by

$$\Sigma_p = \frac{\mathrm{k}_i}{\mathrm{k}_i + \mathrm{k}_j}\Sigma_i + \frac{k_j}{\mathrm{k}_i + \mathrm{k}_j}\Sigma_j \qquad \textbf{(4.53)}$$

and k_i, k_j are the number of inliers in a plane, and Σ_i, Σ_j are the covariance matrices of the plane points.

The orientation and translation tests are already proposed by Weingarten [71] for indoor applications. However, for the outdoor environments the closeness test is crucial. In Figure 4.1, two planes satisfy the first two conditions, namely, their normal vectors seem parallel, and they are almost in the same 2D space. However, they are actually far away from each other and seems belonging to different surfaces. Therefore, the third constraint checks if they are close enough before merging. For that purpose, the Mahalanobis distance with pooled covariance is computed to measure the stochastic distance.

In this study, the threshold parameters are kept constant such that $\mathrm{t}_o = 15°$, $\mathrm{t}_\mathrm{T} = 75\ \mathrm{cm}$, and $\mathrm{t}_\mathrm{P} = 10\ \mathrm{cm}$ throughout the experiments. To be able to propose a good set of threshold parameters, the dataset is analyzed with these values, and they are kept invariable throughout the experiments. In real time implementations, before starting the SLAM, a calibration process, considering the sensor system and environment conditions, is necessary to obtain proper threshold parameters.

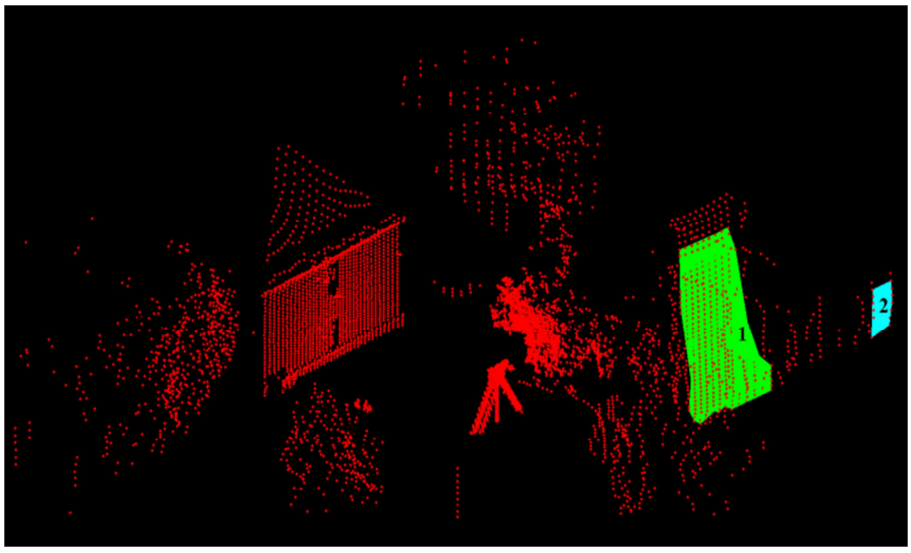


Figure 4.1 : Merging problem.

In this study, an automatic or adaptive method is not investigated for parameter optimization for simplicity but we believe that it may increase the performance of the

feature extraction method. In order to clarify the benefit of using stochastic distance, we have generated three planes and tested these conditions for two cases. In case 1, three planes, described below, lie on the same infinite plane with different ranges as shown in Figure 4.2 (a).

Plane 1: **n**= [0 1 0], *d=N(0,0.2)* bounded within the region of x=0,2,.. 10 and y=0,2,..10.

Plane 2: **n**= [0 1 0], *d=N(0,0.2)* bounded within the region of x=15,16,.. 25 and y=0,1,..10.

Plane 3: **n**= [0 1 0], *d=N(0,0.2)* bounded within the region of x=15,16,.. 45 and y=0,1,..10.

Table 4.1 : Merging tests in two cases.

Cases	Merging Tests Between Planes	C_o	C_t	C_p	d_c
Case 1	1 and 2	4,28	0,19	5,14	15,20
	1 and 3	2,51	0,86	2,38	26,24
Case 2	1 and 2	4,32	4,64	44,77	15,07
	1 and 3	1,62	4,86	141,74	25,88

where d_c is defined as the Euclidean distance $d_c = \sqrt{(c_i - c_j)^T (c_i - c_j)}$ and $N(\mu,\sigma^2)$ represents the normal distributed noise with a mean μ and variance σ^2.

From the Table 4.1, one can see the difference of stochastic distance C_p and Euclidean distances d_c in Case 1. Although the distance of two planes are not changing, the Euclidean distance increases significantly, since it is based on the measure of the center of gravity of two planes. On the other hand, the stochastic distance decreases significantly to force two planes to be merged. In Case 2, the *Plane 2* and *Plane 3* are shifted along the *y*-axis with five units as shown in Figure 4.2 (b). As seen, the Euclidean distances are almost the same with Case 1; however, the stochastic distance increases significantly, since two planes do not belong to the same plane. This shows that the measure of stochastic distance does not only check the closeness of two planes but also check whether two planes are in the same infinite plane or not.

The large finite plane patches are obtained by using this procedure and they are accepted as landmarks if their area is larger than a threshold value.

(a)

(b)

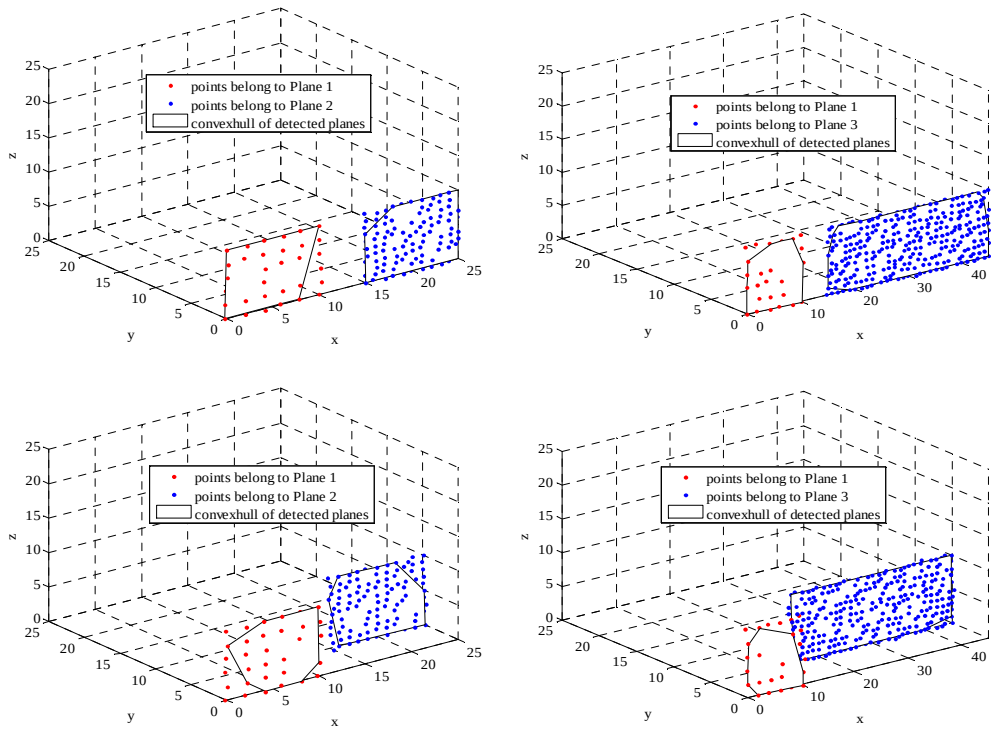


Figure 4.2 : Generated Planes for merging tests: (a)Case 1. (b)Case 2.

As a result, each landmark has the properties of normal vector n_i, center of gravity c_i, convex hull points $\Delta_{XYZ,i}$, covariance matrix of inliers Σ_i, number of inliers k_i, and the plane area A_i, where i is the index of the landmarks. The planar landmarks extracted from the first scan of the Hannover dataset [73] are shown in Figure 4.3. In addition, two triangular planes do not belong to any building walls and roofs, so this shows that the plane-feature based SLAM does not solely depend on the building structures but requires local planar structures.

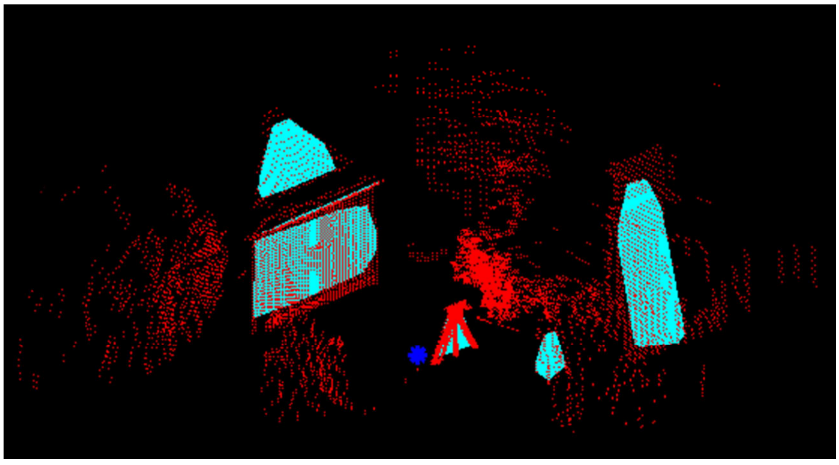


Figure 4.3 : Demonstration of the extracted features.

4.2 Plane-Feature Based SLAM with Gaussian Filters

In this section, the extracted plane features found in the previous section are used in feature based SLAM. The SLAM algorithm uses the infinite plane parameters, which are normal vector and its minimum distance to origin represented in global frame. However, semantic data information of the planes, which are covariance matrix of the plane inliers and plane center of gravity, found in merging step is stored and is used when solving the data association problem. It is started with defining the vehicle, landmark, and sensor models. Then the planar features are adapted to EKF, UKF, CKF, RSPKF, and SEIF based SLAM methods.

Vehicle Model The vehicle model is given by $\mathbf{x}_{v_k} = f(\mathbf{x}_{v_{k-1}}, \mathbf{u}_{v_{k-1}}) + w_{k-1}$ where w_{k-1} denote the zero mean Gaussian distribution noise vector, and **u** is the control signal. Here, f denotes the vehicle model function. In this thesis, the model function f is obtained from the odometry data which is provided by the dataset. Since the odometry provides the rotation and translation information of the vehicle motion, it can be used instead of the kinematic model.

Landmark Model The planar landmarks obtained from the feature extraction are assumed to be stationary $\mathbf{x}_{m_k} = \mathbf{x}_{m_{k+1}}$. These landmarks are augmented in SLAM map with the following state vector representation,

$$x_{m_{k+1}} = \begin{bmatrix} (n_{F_k}^W)^T & d_{F_k}^W \end{bmatrix}^T \tag{4.54}$$

$$\mathbf{x}^a{}_{k+1} = \begin{bmatrix} \mathbf{x}_{v_k}^T & \mathbf{x}_{m_{k+1}}^T \end{bmatrix}^T \in R^{6+4N}. \tag{4.55}$$

Observation Model Sensor measurements, z_k are given by

$$z_k = \begin{bmatrix} (n_{F_k}^L)^T & d_{F_k}^L \end{bmatrix}^T \tag{4.56}$$

where n_F^L and d_F^L are the plane normal and its minimum distance to origin in local frame, respectively. In the formulations, superscripts *W* and *L* indicate the world frame and the local frame, and *T* is the transpose. Since the proposed feature extraction method provides the plane properties in local frame as shown in Figure 4.4, the plane parameters are transformed to the world frame by

$$n_F^W = Rot(x_{v_o}) n_F^L \tag{4.57}$$

$$d_F^W = d_F^L - (n_F^W)^T x_{v_p} \tag{4.58}$$

where Rot matrix represent the three successive rotations defined by the Euler angles of the robot orientation x_{v_o} in *x*, *y*, and *z* axes.

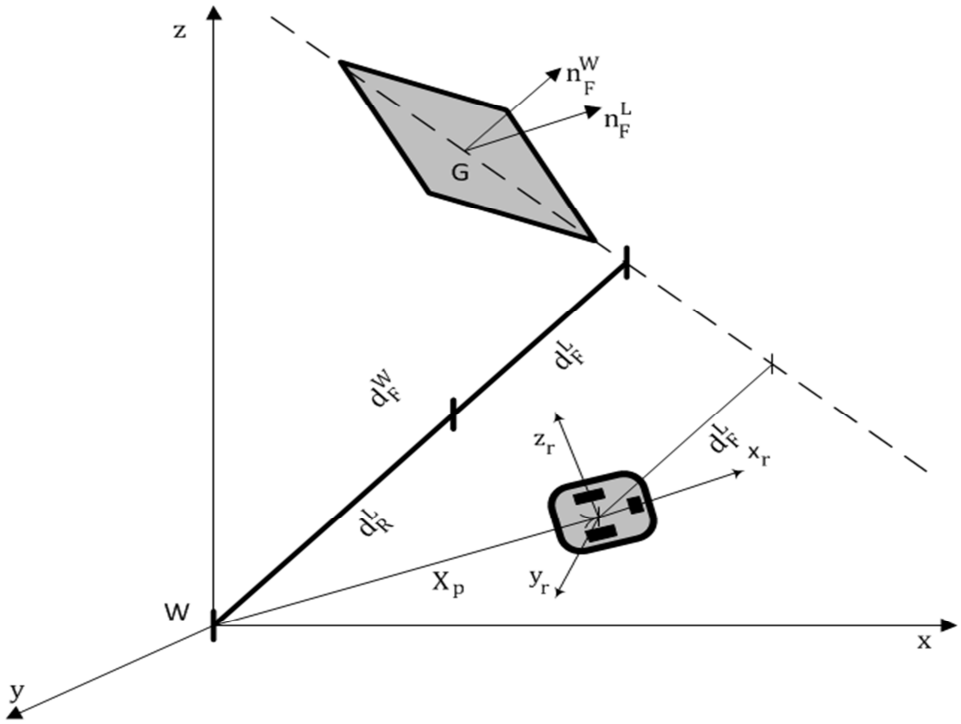


Figure 4.4 : Infinite Plane representation in local and world frame.

The other plane properties such as center of gravity of the planes G_F^L, the covariance matrix C_F^L of the plane points, and convex hull points $\Delta_{xyz,F}^L$ are also transferred to the world frame by using the estimated robot location x_{v_p} and orientation x_{v_o}.

$$C_F^W = Rot(x_{v_o}) C_F^L \tag{4.59}$$

$$G_F^W = Rot(x_o) G_F^L + x_{v_p} \tag{4.60}$$

$$\Delta_{xyz,F}^W = Rot(x_{v_o}) \Delta_{xyz,F}^L + x_{v_p} \tag{4.61}$$

In the next subsection, the EKF SLAM based on proposed plane-feature extraction is introduced.

4.2.1 EKF Based 6D-SLAM with Planar-Features

Kalman filters are the de facto filters for probabilistic SLAM as in many applications. Therefore, the planar-feature extraction method is first adapted to Extended Kalman Filters (EKF). The reason of using the extended version of the Kalman filter is that the motion and observation models are nonlinear functions, and they are approximated to linear with Taylor expansion [51].

To be able to apply the feature extraction method to the EKF based 6D-SLAM method, the Jacobian matrices of the vehicle motion and measurement functions has to be found. Then the traditional EKF time and measurement update procedures are carried out.

Motion Update The motion update depends on finding the Jacobian $\mathbf{F}_{\mathbf{x}}$ of the vehicle model *f* and computed by

$$\mathbf{F}_{\mathbf{x}} = \partial f(\mathbf{x}_{v_k}, \mathbf{u}_{v_k}) / \partial \mathbf{x}_{v_k}. \tag{4.62}$$

Then the Kalman time update step is applied as,

$$\begin{aligned} \mathbf{x}_{v_k}^{-} &= \mathbf{F}_{\mathbf{k}} \mathbf{x}_{v_{k-1}} + u_k \\ \mathbf{P}_k^{-} &= \mathbf{F}_{\mathbf{k}} \mathbf{P}_{k-1} \mathbf{F}_k^T + \mathbf{Q}_{\mathbf{k}}. \end{aligned} \tag{4.63}$$

where $\mathbf{x}_{v_k}^{-}$ represents the predicted state vector and $\mathbf{P}_k^{-}$ is the predicted covariance matrix in the k^{th} step $\mathbf{Q}_{\mathbf{k}}$ denotes the noise covariance matrix of the vehicle motion model.

Measurement Update The sensor measurement model is previously provided by (4.56) and rewritten below for convenience.

$$\begin{aligned} z_k &= h(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) + v_{k-1} \\ &= \begin{bmatrix} n_F^L \\ d_{F_k}^L \end{bmatrix} = \begin{bmatrix} Rot^T(x_{v_o}) n_F^W \\ d_{F_k}^W + (n_F^W)^T x_{v_p} \end{bmatrix} + v_{k-1} \end{aligned} \tag{4.64}$$

The Jacobian matrix of the measurement function H_x is obtained by (4.65) and the EKF measurement update equations are given by (4.66), where $\mathbf{R}_{\mathbf{k}}$ is the covariance

matrix of the measurement noise. Kalman innovation matrix and Kalman gain is denoted by $\mathbf{S}_k$ and $\mathbf{K}_k$, respectively.

$$H_{\mathbf{x}} = \partial h(\mathbf{x}_k, \mathbf{u}_k) / \partial \mathbf{x}_k . \tag{4.65}$$

$$\begin{aligned} \mathbf{S}_k &= \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k \\ \mathbf{K}_k &= \mathbf{P}_k^- \mathbf{H}_k^T \mathbf{S}_k^{-1} \\ \mathbf{x}_{v_k}^+ &= \mathbf{x}_{v_k}^- + \mathbf{K}_k (z_k - \mathbf{H}_k \mathbf{x}_{v_k}^-) \\ \mathbf{P}_k^+ &= (I - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_k^- \end{aligned} \tag{4.66}$$

State Augmentation The projected landmarks are added to the state vector as described in (4.55). The Jacobian matrices of the augmentation function are computed by

$$G_v = \partial x_m / \partial \mathbf{x}_v , \quad G_z = \partial x_m / \partial \mathbf{z} \tag{4.67}$$

and the state covariance matrix is augmented by

$$\begin{aligned} \mathbf{P}_{k,mm} &= G_v \mathbf{P}_{k,vv} G_v^T + G_z R G_z^T \\ \mathbf{P}_{k,vm}^T &= \mathbf{P}_{k,mv} = G_v \mathbf{P}_{k,vv} \\ \mathbf{P}_{k,Lm}^T &= \mathbf{P}_{k,mL} = G_v \mathbf{P}_{k,vL} \end{aligned} \tag{4.68}$$

where *v* represents the vehicle indices in state vector which is 1 to 6, *m* represents the last added landmark indices, and *L* is the whole landmark indices starting from 7 through last index.

4.2.2 UKF Based 6D-SLAM with Planar-Features

Unscented Kalman Filter (UKF) are firstly proposed for nonlinear systems by Julier et al [23]. Since the motion and observation models used in our SLAM representation are nonlinear, UKF is more suitable than EKF since it approximates the nonlinear functions better than EKF. The complexity of the UKF is the same as the EKF but does not require any computation of Jacobians. UKF based SLAM depend on the unscented transform and applied to point feature based SLAM problem for large-scale outdoor SLAM by Martinez and Castellanos [35].

The UKF based 6D-SLAM algorithm with planar-features is given by Algorithm 1 and Algorithm 2 in Table 4.2 and Table 4.3, respectively. Algorithm 1 shows the

prediction and measurement update steps, and the Algorithm 2 explains the state augmentation step. In the prediction step, the state vector is augmented with control signal and sigma points are computed. After propagating the sigma points through vehicle model the state vector and corresponding covariance matrix is predicted. Then in measurement update step, again, the sigma points are evaluated from the predicted values, and they are propagated through the measurement model. Finally, estimated values of the state vector and corresponding covariance matrix are obtained by the Kalman update equations.

EKF requires the computation of Jacobians but since the motion and measurement models are time invariant, they are fixed equations and found by offline. On the other hand, UKF has a vital advantage over EKF if the evaluation of the Jacobian matrix is non-trivial and lead to implementation difficulties [25].

Huang et al. addresses two key limitations of UKF-SLAM problem: the cubic computational complexity related to number of states, and the inconsistency of the state estimates [21]. In particular, they introduce a new sampling strategy minimizing the linearization error with constant computational complexity. They also explore the observability properties of the linear regression based model used by the UKF, and introduce an algorithm, called Observability-Constrained (OC)-UKF, which improves the consistency of the state estimates.

Table 4.2 : Algorithm 1. UKF SLAM with prediction and update steps.

Algorithm 1 $\left(\mathbf{x}_k, \mathbf{P}_k\right) = \mathbf{UKF\_SLAM}\left(\mathbf{x}_{k-1}, \mathbf{P}_{k-1}, \mathbf{u}_{k-1}\right)$

Prediction

1: Augment the state vector and covariance matrix

$$\overline{\mathbf{x}}_{k-1} = \begin{bmatrix} \overline{\mathbf{x}}_{v_{k-1}} & \mathbf{u}_{k-1} \end{bmatrix}^T$$

$$\mathbf{P}_{k-1} = \begin{bmatrix} \mathbf{P}_{k-1} & 0 \\ 0 & Q \end{bmatrix}$$

2: Predict the state vector and covariance matrix

$$[\overline{\mathbf{x}}_k^- \quad \mathbf{P}_k^-] = \mathbf{ukf\_calc}(f(\mathbf{x}_{v_{k-1}}, \mathbf{u}_{v_{k-1}}), \overline{\mathbf{x}}_{k-1}, \mathbf{P}_{k-1})$$

function $[\mathbf{x}, \mathbf{P}] = \mathbf{ukf\_calc}(g, \mathbf{x}, \mathbf{P})$

1: Evaluate the sigma points from $\mathbf{x}$ and $\mathbf{P}$ using unscented transform

$$\chi_i \ (i = 0,1,...,2n)$$

2: Propagate the sigma points through the given model g

$$\mathbf{X}_i = g(\chi_i)$$

3: Estimate the state vector and covariance matrix

$$\mathbf{x} = \sum_{i=0}^{2n} W_i^m \mathbf{X}_i$$

$$\mathbf{P} = \sum_{i=0}^{2n} W_i^c \mathbf{X}_i \mathbf{X}_i^T - \mathbf{x}\mathbf{x}^T$$

Measurement Update

1: Evaluate the sigma points using unscented transform

$$\chi_{i,k}^- = \bar{\mathbf{x}}_k^- \qquad (i = 0)$$

$$\chi_{i,k}^- = \bar{\mathbf{x}}_k^- + \left\{\sqrt{(n+\lambda)\mathbf{P}_k^-}\right\}_i \qquad (i = 1,2,...,n)$$

$$\chi_{i,k}^- = \bar{\mathbf{x}}_k^- - \left\{\sqrt{(n+\lambda)\mathbf{P}_k^-}\right\}_i \qquad (i = n+1,...,2n)$$

2: Propagate the sigma points in observation model and predict the measurement

$$\mathbf{Z}_{i,k}^- = \mathbf{h}(\chi_{i,k}^-, \mathbf{u}_k)$$

$$\bar{\mathbf{z}}_k^- = \sum_{k=0}^{2n} W_i^m \mathbf{Z}_{i,k}^-$$

3: Estimate innovative and the cross covariance matrices

$$\mathbf{P}_{zz,k}^- = \sum_{k=0}^{2n} W_k^c (\mathbf{Z}_{i,k}^- - \bar{\mathbf{z}}_k^-)(\mathbf{Z}_{i,k}^- - \bar{\mathbf{z}}_k^-)^T + R_k$$

$$\mathbf{P}_{xz,k}^- = \sum_{k=0}^{2n} W_k^c (\mathbf{X}_{i,k}^- - \bar{\mathbf{z}}_k^-)(\mathbf{X}_{i,k}^- - \bar{\mathbf{z}}_k^-)^T$$

4: Estimate the updated state and covariance matrix

$$\mathbf{K}_k = \mathbf{P}_{xz,k}^- (\mathbf{P}_{zz,k}^-)^{-1}$$

$$\bar{\mathbf{x}}_k^+ = \bar{\mathbf{x}}_k^- + \mathbf{K}_k (\mathbf{z}_k - \bar{\mathbf{z}}_k^-)$$

$$\mathbf{P}_k^+ = \mathbf{P}_k^- - \mathbf{K}_k \mathbf{P}_{zz,k}^- \mathbf{K}_k^T$$

Table 4.3 : Algorithm 2. UKF state augmentation.

Algorithm 2 $[\mathbf{x}_k^a \quad \mathbf{P}_k^a] = \mathbf{State\_Augmentation}\left(\mathbf{x}_k, \mathbf{P}_k\right)$

Augmentation

1: Augment the state vector and covariance matrix

$$\mathbf{x}_k = \begin{bmatrix} x_k & z_k \end{bmatrix}^T$$

$$\mathbf{P}_k = \begin{bmatrix} \mathbf{P}_k & 0 \\ 0 & R \end{bmatrix}$$

2: Evaluate the state vector and covariance matrix

$$g_k^a = \begin{bmatrix} \mathbf{x}_{v_k} & x_{m_k} \end{bmatrix}^T$$

$$\begin{bmatrix} \mathbf{x}_k^a & \mathbf{P}_k^a \end{bmatrix} = \mathbf{ukf\_calc}(g_k^a, \mathbf{x}_k, \mathbf{P}_k)$$

4.2.2.1 Cubature Theory

The key point of the Cubature theory [1] is to find multi-dimensional integrals using cubature rules since its integrands are in the form of *nonlinear function* × *Gaussian Distribution*. Thus, the Bayesian filter solution is approximated by the help of cubature theory.

Cubature Rule The cubature rule is used to approximate an *n*-dimensional Gaussian weighted integral as

$$\int_{R^n} f(x)N(x;\mu,P)dx \approx \frac{1}{2n}\sum_{i=1}^{2n} f(\mu + \zeta_i\sqrt{P}) \qquad \textbf{(4.69)}$$

where *N* is the normal distribution of x with mean, μ, and covariance matrix, P. The relation for covariance matrix $P = \sqrt{P}\sqrt{P}^T$ is satisfied. The *2n* set of cubature array set is defined by ζ, and ζ_i the i^{th} element of the set ζ,

$$\zeta = \sqrt{n}\left\{\begin{pmatrix}1\\.\\.\\.\\0\end{pmatrix},\ldots,\begin{pmatrix}0\\.\\.\\.\\1\end{pmatrix},\begin{pmatrix}-1\\.\\.\\.\\0\end{pmatrix},\ldots,\begin{pmatrix}0\\.\\.\\.\\-1\end{pmatrix}\right\}. \qquad \textbf{(4.70)}$$

4.2.2.2 Feature Based CKF SLAM

The usage of extracted features in CKF SLAM is explained in this section. Feature based CKF-SLAM algorithm uses the infinite plane parameters as features. However, other semantic data information of the planes found in landmark extraction is used when solving the data association problem. Thus, every plane feature is considered as an infinite plane in SLAM posterior; however, in fact they are plane patches in the correspondence decision. First, the vehicle, landmark, and the sensor models are elucidated as follows. Then CKF-SLAM motion and measurement update steps are explained.

Vehicle Model The vehicle model is given by

$$\mathbf{x}_{vk} = f(\mathbf{x}_{v_{k-1}}, \mathbf{u}_{k-1}) + w_{k-1} \tag{4.71}$$

where w_{k-1} denotes the zero mean Gaussian distribution noise vector with the covariance matrix Q, and the control signal or odometry data is provided as $\mathbf{u} = [\delta x, \delta y, \delta z, \delta\alpha, \delta\beta, \delta\gamma]$. The vehicle state vector is represented by $\mathbf{x}_v = [\mathbf{x}_{v_p} \quad \mathbf{x}_{v_o}]$ where $\mathbf{x}_{v_p} = [x, y, z]$ and $\mathbf{x}_{v_o} = [\alpha, \beta, \gamma]$ denote the robot position and the orientation, respectively. The vehicle model function given by *f* and can be disclosed as in (4.71) for the odometry data having the relative rigid body transformation parameters.

$$\begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ z_{k-1} \end{bmatrix} + Rot(x_{v_{o,k}}) \begin{bmatrix} \delta x_{k-1} \\ \delta y_{k-1} \\ \delta z_{k-1} \end{bmatrix}$$
$$\begin{bmatrix} \alpha_k \\ \beta_k \\ \gamma_k \end{bmatrix} = \begin{bmatrix} \alpha_{k-1} \\ \beta_{k-1} \\ \gamma_{k-1} \end{bmatrix} + \begin{bmatrix} \delta\alpha_{k-1} \\ \delta\beta_{k-1} \\ \delta\gamma_{k-1} \end{bmatrix} \tag{4.72}$$

where *Rot* matrix represent the three successive rotations defined by the Euler angles in x, y, and z axes. The motion and measurement update steps of the CKF SLAM are explained as follows.

Motion Update Motion update step is based on the vehicle model (4.71). The state and covariance matrix is augmented as

$$\bar{\mathbf{x}}_{k-1} = \begin{bmatrix} \bar{\mathbf{x}}_{v_{k-1}} & \mathbf{u}_{k-1} \end{bmatrix}^T$$
$$\mathbf{P}_{k-1} = \begin{bmatrix} \mathbf{P}_{k-1} & 0 \\ 0 & Q \end{bmatrix} \quad \textbf{(4.73)}$$

where $\bar{\mathbf{x}}_{k-1}$ is the state vector in the (k-1)th time step and $\mathbf{u}_{k-1}$ is the applied control signal at this time. $\mathbf{P}_{k-1}$ denotes the state covariance matrix. The cubature array, ζ, is evaluated with (4.70), and the square root of the P matrix is obtained with Cholesky decomposition, $\Sigma = chol(P)^T$. For the sake of simplicity, the time index (*k*) is reduced from the following equations. Cubature points are evaluated by

$$\boldsymbol{\kappa}_i^- = \bar{\mathbf{x}} + \sum_{i=1}^{2n} \zeta_i \quad \textbf{(4.74)}$$

Then *2n* cubature points are propagated through the vehicle model as follows,

$$\mathbf{X}_i = f(\boldsymbol{\kappa}_i^-, \mathbf{u}). \quad \textbf{(4.75)}$$

The estimated state and corresponding state vector is obtained by

$$\bar{\mathbf{x}}^- = \frac{1}{2n} \sum_{i=1}^{2n} \mathbf{X}_i$$
$$\mathbf{P} = \frac{1}{2n} \sum_{i=1}^{2n} \mathbf{X}_i \mathbf{X}_i^T - (\bar{\mathbf{x}}^-)(\bar{\mathbf{x}}^-)^T. \quad \textbf{(4.76)}$$

After the motion update, measurement update step is applied as follows.

Measurement Update The measurement update step is based on the observation model (4.64). The cubature points are evaluated by the estimated state and covariance matrix as $\boldsymbol{\kappa}_i^+ = \bar{\mathbf{x}}^- + \sqrt{\mathbf{P}}\zeta_i$, and then they are propagated through the observation model as

$$Z_i = h(\boldsymbol{\kappa}_i^+, \mathbf{u}_{k-1}) + v_{k-1}. \quad \textbf{(4.77)}$$

where v_{k-1} is the zero mean measurement Gaussian noise with *R* covariance matrix. The predicted measurement mean vector is computed as

$$\bar{\mathbf{z}}_i^- = \frac{1}{2n}\sum_{i=1}^{2n} Z_i \tag{4.78}$$

and the cross covariance matrices are obtained by

$$\begin{aligned}\mathbf{P}_{zz} &= \frac{1}{2n}\sum_{i=1}^{2n} Z_i Z_i^T - (\bar{\mathbf{z}}_i^-)(\bar{\mathbf{z}}_i^-)^T + R \\ \mathbf{P}_{xz} &= \frac{1}{2n}\sum_{i=1}^{2n} X_i Z_i^T - (\bar{\mathbf{x}}^-)(\bar{\mathbf{z}}_i^-)^T\end{aligned} \tag{4.79}$$

The Kalman Gain (G), updated state vector and covariance matrix are evaluated by

$$\begin{aligned}\mathbf{G} &= \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1} \\ \bar{\mathbf{x}}^+ &= \bar{\mathbf{x}}^- + \mathbf{G}(z - \bar{z}) \\ \mathbf{P} &= \mathbf{P} - \mathbf{G}\mathbf{P}_{zz}\mathbf{G}^T\end{aligned} \tag{4.80}$$

If a new landmark is observed, it is added to the state vector with the state augmentation model.

State Augmentation The state augmentation is based on the landmark model definition (4.54) and the augmentation given by (4.55). The state augmentation is applied in two steps. Firstly, the state vector and covariance matrix is augmented with the new observations as follows.

$$\begin{aligned}\bar{\mathbf{x}}^+ &= \begin{bmatrix}\bar{\mathbf{x}}_v & z\end{bmatrix}^T \\ \mathbf{P} &= \begin{bmatrix}\mathbf{P} & 0 \\ 0 & R\end{bmatrix}\end{aligned} \tag{4.81}$$

The augmented state model $g^a = \begin{bmatrix}\mathbf{x}_v & x_m\end{bmatrix}$ is constructed, and then the augmented state vector and covariance matrix are computed by following the same procedure in motion update step with (4.74) , (4.75), and (4.76).

4.2.3 RSPKF Based 6D-SLAM with Planar-Features

The sigma point filters provides an approximate solutions to the nonlinear functions. However, these approximations are biased, and they generate systematic errors. To keep the equations more certain, we elucidate this situation on the Unscented Transform (UT). The error is expressed by means of the Taylor expansion of the actual mean $\bar{\mathbf{y}}$ and the approximate mean $\bar{\mathbf{y}}^{ut}$.

The Taylor expansion of the true mean of *y* can be written as

$$\bar{\mathbf{y}} = \mathbf{g}(\bar{\mathbf{x}}) + \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{P(i,j)}{2}\nabla_{i,j}^{2}\mathbf{g} + E\left[\frac{D_{\Delta x}^{4}\mathbf{g}(\mathbf{x})}{4!} + \frac{D_{\Delta x}^{6}\mathbf{g}(\mathbf{x})}{6!} + ..\right] \tag{4.82}$$

where

$$\nabla_{i,j}^{2}\mathbf{g} = \frac{\partial^{2}}{\partial x_i \partial x_j}\mathbf{g}(\mathbf{x})\,|_{x=\bar{x}}, \quad \frac{D^{k}\mathbf{g}(\mathbf{x})}{k!} = \frac{1}{k!}\left(\sum\nolimits_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}}_i)\frac{\partial}{\partial x_i}\right)^{k}\mathbf{g}(\mathbf{x})\,|_{x=\bar{x}} \tag{4.83}$$

Based on the sigma points and the corresponding weights the approximate mean is written by the Julier et al. [24] as follows

$$\begin{aligned}\bar{\mathbf{y}} = \mathbf{g}(\bar{\mathbf{x}}) &+ \frac{n+\kappa}{2(n+\kappa)}\sum_{i=1}^{n}\sum_{j=1}^{n}P(i,j)\nabla_{i,j}^{2}\mathbf{g} \\ &+ \frac{1}{2(n+\kappa)}\sum_{p=1}^{2n}\left(\frac{D_{\varepsilon_p}^{4}\mathbf{g}(\chi_p)}{4!} + \frac{D_{\varepsilon_p}^{6}\mathbf{g}(\chi_p)}{6!} + \ldots\right)\end{aligned} \tag{4.84}$$

where

$$\frac{D_{\varepsilon_p}^{k}\mathbf{g}(\mathbf{x})}{k!} = \frac{1}{k!}\left(\sum\nolimits_{i=1}^{n}(\chi_p(i) - \bar{\mathbf{x}}_i)\frac{\partial}{\partial \chi_p(i)}\right)^{k}\mathbf{g}(\chi_p)\,|_{\chi_p=\bar{x}} \tag{4.85}$$

is the k^{th} term of the Taylor series expansion of the p^{th} sigma point $\mathbf{g}(\chi_p)$ and $\chi_p(i)$ is the i^{th} element of χ_p.

The systematic error can be written as

$$\begin{aligned}\varepsilon^{ut} &= E\left[\frac{D_{\Delta x}^{4}\mathbf{g}(\mathbf{x})}{4!} + \frac{D_{\Delta x}^{6}\mathbf{g}(\mathbf{x})}{6!} + \ldots\right] \\ &- \frac{1}{2(n+\kappa)}\sum_{p=1}^{2n}\left(\frac{D_{\varepsilon_p}^{4}\mathbf{g}(\chi_p)}{4!} + \frac{D_{\varepsilon_p}^{6}\mathbf{g}(\chi_p)}{6!} + \ldots\right)\end{aligned} \tag{4.86}$$

The error ε^{ut} is different from zero if the function g is not a polynomial of degree *2n*. This systematic error is also appears in the computations of the covariance matrices in a similar fashion. The randomized sigma point Kalman filter is proposed to eliminate the mentioned systematic error. Randomized Sigma Point Kalman Filter (RSPKF) proposed by Dunik et al. [12] uses the stochastic integration rule (SIR) introduced by Genz and Monohan [18]. The SIR is summarized in the next subsection.

4.2.3.1 Stochastic Integration Rule

SIR is appropriate for solving the integral of the form

$$\mu = \int_{\square^n} \mathbf{g}(\mathbf{x}) \left(\frac{1}{2\pi}\right)^{n/2} \mathrm{e}^{-\frac{1}{2}\mathbf{x}^T\mathbf{x}} \qquad \textbf{(4.87)}$$

This relation can be considered as a computation of the expected value of the function g where x is a random variable with $p(\mathbf{x}) = N(\mathbf{x}; \bar{\mathbf{x}}, \mathbf{P})$. The algorithm to solve the integral based on SIR is given by the Algorithm 3 in Table 4.4. The matrix **Q** can be generated using a product of appropriately chosen random reflections [18]. It is claimed in the paper that the SIR provides superior performance with respect to conventional methods in cases where the function **g(x)** is not approximately constant.

Table 4.4 : Algorithm 3. Stochastic Integration Rule.

Algorithm 3 $\mu = \mathbf{SI}(\bar{\mathbf{x}}, \mathbf{P}, \mathbf{g(x)})$

1: Define N_{max}

2: Set $\mu = \mathbf{0}$ and compute $\chi_0 = \mathbf{g}(\bar{\mathbf{x}})$

3: **for** i=1 to N_{max} do

4: Generate a uniformly random orthogonal matrix $\mathbf{Q} \in R^{nxn}$ and generate a random number ρ form Chi-distribution with $n+2$ degrees of freedom.

5: Compute a set of points $\boldsymbol{\chi}_i$ and corresponding weights w_i from

$$\boldsymbol{\chi}_i = -\rho\sqrt{P}\mathbf{Q}\mathbf{e}_i$$

$$\boldsymbol{\chi}_{n+i} = \rho\sqrt{P}\mathbf{Q}\mathbf{e}_i$$

$$w_0 = 1 - \frac{n}{\rho^2},\ w_i = w_{n+i} = \frac{1}{2\rho^2}$$

where $i=1,2\ldots n$ and $\mathbf{e}_i$ is the i^{th} column of the identity matrix.

6: Compute the value **S** of the integral at current iteration

$$\mathbf{S} = -\boldsymbol{\chi}_0\omega_0 + \sum_{i=0}^{n_x}\left(\mathbf{g}(\boldsymbol{\chi}_i) + \mathbf{g}(\boldsymbol{\chi}_{i+n_x})\right)\omega_i$$

and use it to update the approximate mean μ

$$\mu = \mu + (\mathbf{S} - \mu)/i$$

7: **end for**

8: **return** μ

4.2.3.2 Randomized Sigma Point Kalman Filter

The time update and the measurement update steps are based on the SIR algorithm and are given as follows.

Time Update RSPKFs computes the predicted mean $\bar{\mathbf{x}}^-$ and covariance matrix $\mathbf{P}^-$ depending on the transformation.

$$\bar{\mathbf{x}}_k^- = E[\, f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) + w_{k-1} \mid D_{k-1}] \tag{4.88}$$

where D_{k-1} denotes the history of the input and measurement pairs up to k-1. Since w_{k-1} is assumed to be zero mean and independent of the measurement sequence, one can write

$$\begin{aligned} \bar{\mathbf{x}}_k^- &= E[\, f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) \mid D_{k-1}\,] \\ &= \int_{R^n} f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1})\, p(\mathbf{x}_{k-1} \mid D_{k-1})\, d\mathbf{x}_{k-1} \\ &= \int_{R^n} f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) N(\mathbf{x}_{k-1}; \bar{\mathbf{x}}_{k-1}, \mathbf{P}_{k-1})\, d\mathbf{x}_{k-1} \end{aligned} \tag{4.89}$$

The corresponding error covariance matrix can be written as

$$\begin{aligned} \mathbf{P}_k^- &= E[(\mathbf{x}_k - \bar{\mathbf{x}}_k^-)(\mathbf{x}_k - \bar{\mathbf{x}}_k^-)^T \mid \mathrm{z}_{k-1}\,] \\ &= \int_{R^n} f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1})\, f^T(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) \\ &\quad x\, N(\mathbf{x}_{k-1}; \bar{\mathbf{x}}_{k-1}, \mathbf{P}_{k-1}) d\mathbf{x}_{k-1} - \bar{\mathbf{x}}_{k-1} \bar{\mathbf{x}}_{k-1}^T + Q_{k-1} \end{aligned} \tag{4.90}$$

Here the integrals are solved by the SIR algorithm.

$$\begin{aligned} \bar{\mathbf{x}}_k^- &= \mathbf{SI}\left(\bar{\mathbf{x}}_k, \mathbf{P}_k, f_k(\mathbf{x}_k)\right) \\ \mathbf{P}_k^- &= \mathbf{SI}\left(\bar{\mathbf{x}}_k, \mathbf{P}_k, (f_k(\mathbf{x}_k) - \bar{\mathbf{x}}_k)(f_k(\mathbf{x}_k) - \bar{\mathbf{x}}_k)^T\right) + Q_k \end{aligned} \tag{4.91}$$

Measurement Update The predicted measurements vector, the corresponding covariance and cross covariance matrices are given by

$$\begin{aligned} \bar{\mathbf{z}}_k^- &= \int_{R^n} h(\mathbf{x}_k) N(\mathbf{x}_k, \bar{\mathbf{x}}_{k-1}, \mathbf{P}_{k-1})\, d\mathbf{x}_k \\ P_{zz,k} &= \int_{R^n} h(\mathbf{x}_k) h^T(\mathbf{x}_k) N(\mathbf{x}_k; \bar{\mathbf{x}}_{k-1}, \mathbf{P}_{k-1}) d\mathbf{x}_k - \bar{\mathbf{z}}_k \bar{\mathbf{z}}_k^T + R_k \\ P_{xz,k} &= \int_{R^n} \mathbf{x}_k h^T(\mathbf{x}_k) N(\mathbf{x}_k; \bar{\mathbf{x}}_{k-1}, \mathbf{P}_{k-1}) d\mathbf{x}_k - \bar{\mathbf{x}}_{k-1} \bar{\mathbf{z}}_k^T \end{aligned} \tag{4.92}$$

The integrals in the measurement step are obtained by the SIR algorithm by

$$\begin{aligned} \bar{\mathbf{z}}_k^- &= \mathbf{SI}\left(\bar{\mathbf{x}}_k^-, \mathbf{P}_k^-, h_k(\mathbf{x}_k)\right) \\ P_{zz,k} &= \mathbf{SI}\left(\bar{\mathbf{x}}_k^-, \mathbf{P}_k^-, (h_k(\mathbf{x}_k) - \bar{\mathbf{z}}_k^-)(h_k(\mathbf{x}_k) - \bar{\mathbf{z}}_k^-)^T\right) + \mathbf{R}_k \\ P_{xz,k} &= \mathbf{SI}\left(\bar{\mathbf{x}}_k^-, \mathbf{P}_k^-, (\mathbf{x}_k - \bar{\mathbf{x}}_k^-)(h_k(\mathbf{x}_k) - \bar{\mathbf{z}}_k^-)^T\right) + \mathbf{R}_k. \end{aligned} \tag{4.93}$$

After new $\mathbf{z}_k$ measurements are obtained, the sigma point Kalman filter updates the state vector and covariance matrix as

$$\bar{\mathbf{x}}_k^+ = \bar{\mathbf{x}}_k^- + K_k(\mathbf{z}_k - \bar{\mathbf{z}}_k)$$
$$P_k = P_k^- - K_k P_{zz,k} K_k^T \quad \textbf{(4.94)}$$

where the K_k is the Kalman gain given by

$$K_k = P_{xz,k} P_{zz,k}^{-1} \quad \textbf{(4.95)}$$

4.2.3.3 SLAM Based on RSPKF

The conventional feature based SLAM representation consists of three models, which are vehicle model *f*, observation model *h*, and the augmentation model *g*. Although these representations given in the previous sections, they rewritten here for consistency.

Vehicle Model The vehicle model is given by

$$\mathbf{x}_{v_k} = f(\mathbf{x}_{v_{k-1}}, \mathbf{u}_{k-1}) + w_{k-1} \quad \textbf{(4.96)}$$

where w_{k-1} denotes the zero mean Gaussian distribution noise vector with the covariance matrix **Q**, and the control signal **u.**

Landmark Model The landmarks are assumed as stationary $\mathbf{x}_{m_k} = \mathbf{x}_{m_{k+1}}$ and represented in world (*W*) frame. The SLAM map is augmented with the following state vector representation,

$$\mathbf{x}^a{}_{k+1} = \begin{bmatrix} \mathbf{x}_{v_k} & \mathbf{x}_{m_{k+1}} \end{bmatrix} \in \mathrm{R}^{6+4\mathrm{N}}. \quad \textbf{(4.97)}$$

Observation Model Measurement or observation model parameters, z_k are provided by the feature extraction method and stated as

$$z_k = h(\mathbf{x}, \mathbf{u}_{k-1}) + v_{k-1} \quad \textbf{(4.98)}$$

where h is the measurement model and v_{k-1} is the zero mean observation noise with R error covariance matrix.

Motion Update Motion update step is based on the vehicle model (4.72). The state and covariance matrix is augmented as

$$\bar{\mathbf{x}}_k = \begin{bmatrix} \bar{\mathbf{x}}_{v_k} & \mathbf{u}_k \end{bmatrix}^T$$
$$\mathbf{P}_k = \begin{bmatrix} \mathbf{P}_k & 0 \\ 0 & Q_k \end{bmatrix} \quad \textbf{(4.99)}$$

where $\bar{\mathbf{x}}_k$ is the state vector in the k^{th} time step and is the applied control signal at this time. $\mathbf{P}_k$ denotes the state covariance matrix and augmented as in (4.99). The square root S of the covariance matrix $\mathbf{P}_k$ is obtained by the Cholesky decomposition S=*chol*($\mathbf{P}_k$). Then the time prediction step of the RSPKF algorithm is applied to the augmented vectors as

$$\bar{\mathbf{x}}_k^- = \mathbf{SI}\left(\bar{\mathbf{x}}_k, \mathbf{P}_k, f_k(\mathbf{x}_k)\right)$$
$$\mathbf{P}_k^- = \mathbf{SI}\left(\bar{\mathbf{x}}_k, \mathbf{P}_k, (f_k(\mathbf{x}_k) - \bar{\mathbf{x}}_k)(f_k(\mathbf{x}_k) - \bar{\mathbf{x}}_k)^T\right) + Q_k. \quad \textbf{(4.100)}$$

Measurement Update The measurement update step is based on the observation model (4.64) and the state and covariance estimations are obtained using the SIR as in (4.93).

$$\bar{\mathbf{x}}_k^+ = \bar{\mathbf{x}}_k^- + K_k(\mathbf{z}_k - \bar{\mathbf{z}}_k)$$
$$\mathbf{P}_k = \mathbf{P}_k^- - K_k \mathbf{P}_{zz,k} K_k^T \quad \textbf{(4.101)}$$

where the K_k is the Kalman gain given by

$$K_k = \mathbf{P}_{xz,k} \mathbf{P}_{zz,k}^{-1} \quad \textbf{(4.102)}$$

State Augmentation The state augmentation is based on the augmentation given by (4.97) and operated in every new landmark observations. The state augmentation is applied in two steps. Firstly, the state vector and covariance matrix is augmented with the new observations as follows.

$$\bar{\mathbf{x}}_k^+ = \begin{bmatrix} \bar{\mathbf{x}}_{v_k} & \mathbf{z}_k \end{bmatrix}^T$$
$$\mathbf{P}_k = \begin{bmatrix} \mathbf{P}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_k \end{bmatrix} \quad \textbf{(4.103)}$$

The augmented state model $g^a = \begin{bmatrix} \mathbf{x}_v & x_m \end{bmatrix}$ is constructed, and then the augmented state vector and covariance matrix are computed by following the same procedure in motion update step with (4.99) and (4.100).

4.2.4 SEIF Based 6D-SLAM with Planar-Features

The classical filtering methods like EKF and UKF for solving SLAM problem require quadratic update time depending on the number landmarks in map. For that reason, a scalable algorithm is necessary for large scale SLAM, where its update time does not depend on the number landmarks. Recently, Sparse Extended Information Filters (SEIF) are proposed for this purpose by representing maps through local, web-like networks of features [56]. Thus, SLAM updates can be performed in constant time without regard to number of landmarks in the map. In addition, due to the nature of spares structure, SEIF filters based SLAM requires less memory storage than the mentioned Kalman filters based SLAM. Moreover, since the information form, which is canonical representation of the covariance form, is more suitable for multiple platforms and multi-sensor fusion frameworks, an advantageous method for long range SLAM is obtained. Before explaining the sparse extended information filters, the fundamental part of the method, which is the extended information filters are summarized in the next subsection.

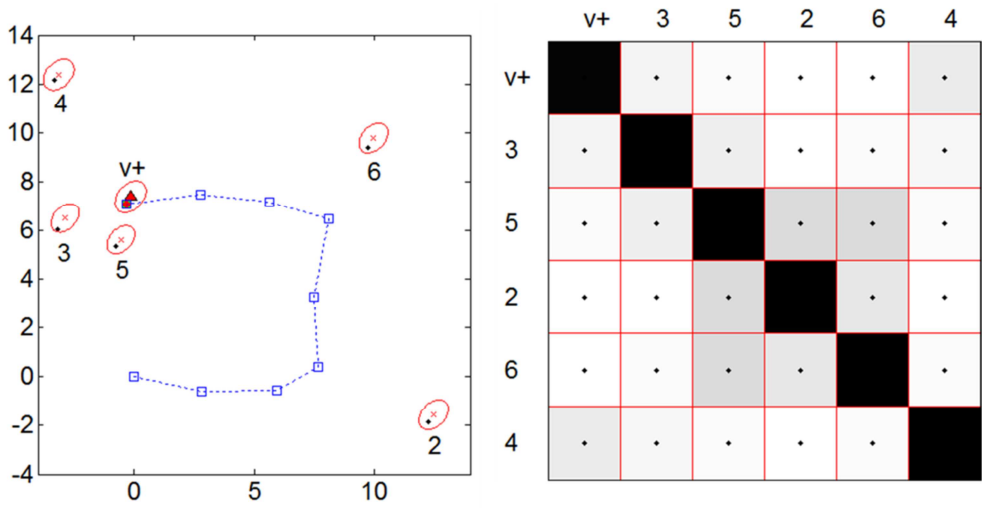


Figure 4.5 : Representation of the robot and feature positions: (a)Features and robot positions with their uncertainty ellipsoids. (b)The information Matrix.

4.2.4.1 Extended Information Filters

Extended Information Filter (EIFs) forms the basis of the Sparse Information Filters (SEIF) and are computationally equivalent to EKFs. However, they represent the information in a canonical form: instead of using the covariance matrix, the EIF uses the inverse covariance matrix called information matrix. Similar to EKF, the word “extend” represent the extension of the filter to non-linear form. For consistency, we

use the same notation as in the original paper. An illustrative simulation result is shown in Figure 4.5, where the normalized information matrix is naturally sparse and the relations (links) close to diagonal axis are more strength than that of the off-diagonal elements. In this figure, it is also shown that the state vector is augmented in the order of the observations and the brightness indicates low values and *dot*s show the non-zero but small values. The EIF for feature based SLAM in terms of probabilistic representation is explained as follows.

The SLAM posterior $p(\xi_t, z^t, u^t)$ is conditioned on the past sensor measurements $z^t = z_1, ..., z_t$ and past controls $u^t = u_1, ..., u_t$. The multivariate normal distribution over the state vector ξ_t is given in (4.104) with the mean of this distribution μ_t and covariance matrix Σ_t.

$$p(\xi_t, z^t, u^t) \propto \exp\left\{-\frac{1}{2}(\xi_t - \mu_t)^T \Sigma_t^{-1}(\xi_t - \mu_t)\right\} \quad \textbf{(4.104)}$$

The information filter represents the same posterior with *information matrix* H_t and *information vector* b_t as follows,

$$p(\xi_t, z^t, u^t) \propto \exp\left\{-\frac{1}{2}\xi_t^T H_t \xi_t + b_t \xi_t\right\} \quad \textbf{(4.105)}$$

where the following relation exists between the covariance and information forms.

$$H_t = \Sigma_t^{-1} \quad \text{and} \quad b_t = \mu^T H_t \quad \textbf{(4.106)}$$

The information matrix H_t is symmetric and positive-definite and each element in it constraints one (on the main diagonal) or two (off the main diagonal) elements in the state vector. The off-diagonal elements in H_t matrix are considered as *links*, which means H_{x_t, y_n} links the robot pose estimate and the specific feature location estimate.

Measurement Update The measurement update of the EIF is given by the additive rule as

$$\begin{aligned} H_t &= \bar{H}_t + C_t Z^{-1} C^T \\ b_t &= \bar{b}_t + \left(z_t - \bar{z}_t + C_t^T \mu\right) Z^{-1} C_t^T \end{aligned} \quad \textbf{(4.107)}$$

where Z is the covariance matrix of the noise variable (zero mean) on the measurement and the Jacobian C_t is given in (4.108) which is a sparse matrix since it only have non-zero elements corresponding to the robot pose x_t and the feature y_t observed at time *t*. The reason of the sparseness is that the measurements z_t are only function of the relative distance and orientation of the robot to the observed feature.

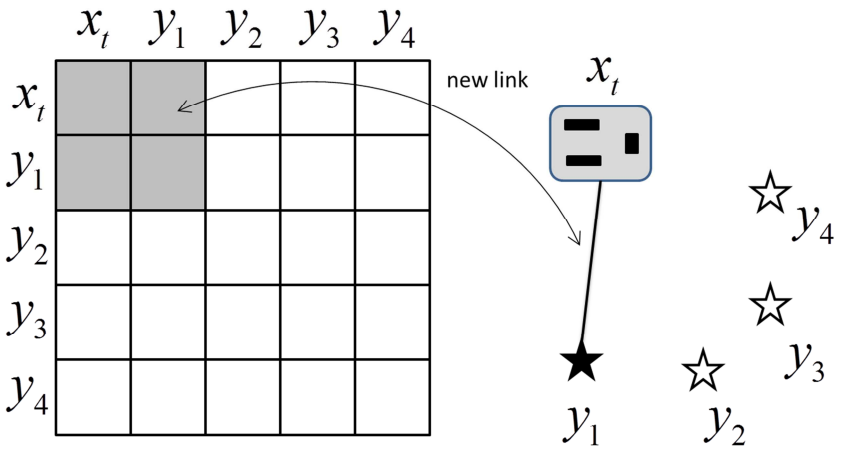


Figure 4.6 : The effect of a first measurement update in the information matrix.

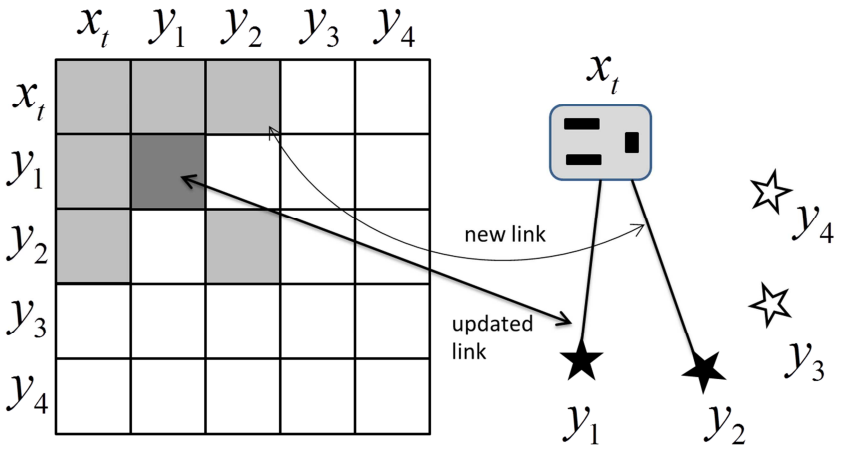


Figure 4.7 : The effect of a second measurement update in the information matrix.

In Figure 4.6, mobile robot observes the feature y_1, which makes a modification on the information matrix elements H_{x_t,y_1}, and in Figure 4.7, the robot observes the second feature y_2, which particularly affects H_{x_t,y_2}. In Figure 4.7, it is also shown that, the reobvserved feature y_1 increases its stregth through measurment update. The key point of EIF is that this measurment update is performed in constant time due to the nature of the information filters.

$$C_t = \left(\frac{\partial h}{\partial x_t} \quad 0 \cdots 0 \quad \frac{\partial h}{\partial y_t} \quad 0 \cdots 0 \right)^T \tag{4.108}$$

The motion update step of the EIF is given below.

Motion Update The second step of SLAM is the update of the filter because of the robot motion. The effect of the robot motion on the information matrix is more complicated than that of measurements. Since the motion is non-deterministic, the motion update introduce new links or enhance existing links between any two active features, while weakens the links between the robot and those features as shown in Figure 4.8. This is a consequence of marginalization of robot pose, and the details are discussed in [56]. The motion update equations in terms of information form are given as

$$\begin{aligned} H_t = \bar{\Sigma}_t^{-1} &= \left[(I + A_t) \Sigma_{t-1} (I + A_t)^T + S_x U_t S_x^T \right]^{-1} \\ &= \left[(I + A_t) H_{t-1}^{-1} (I + A_t)^T + S_x U_t S_x^T \right]^{-1} \\ b_t = \bar{\mu}^T \bar{H}_t &= \left[\mu_{t-1} + \Delta_t \right]^T \bar{H}_t = \left[H_{t-1}^{-1} b_{t-1}^T + \Delta_t \right]^T \bar{H}_t \\ &= \left[b_{t-1} H_{t-1}^{-1} + \Delta_t^T \right]^T \bar{H}_t \end{aligned} \tag{4.109}$$

where Δ_t is the predicted motion effect and S_x is the projection matrix of the form $S_x = (I \ 0 \ldots 0)^T$,where I is the identity matrix of the same dimension as the robot pose vector. U_t denotes the covariance matrix of the zero mean noise variable on the motion of the robot. A_t is the Jacobian of the vehicle model function.

4.2.4.2 Sparse Extended Information Filters

In general, the complexity of the EIF is cubic in the size of the state space. Therefore, the number of landmarks increases the computational complexity of the EIF filters as in EKF. However, if the information matrix is sparse or can be made sparse, the update time is achieved in constant time. The results of the SEIF are given with three lemmas below and the sparsification approach is explained in the next part.

Lemma 1: The measurement update requires constant time, irrespective of the number of landmarks in the state vector. This lemma does not require sparseness of the information matrix and it is a well-known property of the information filters in SLAM.

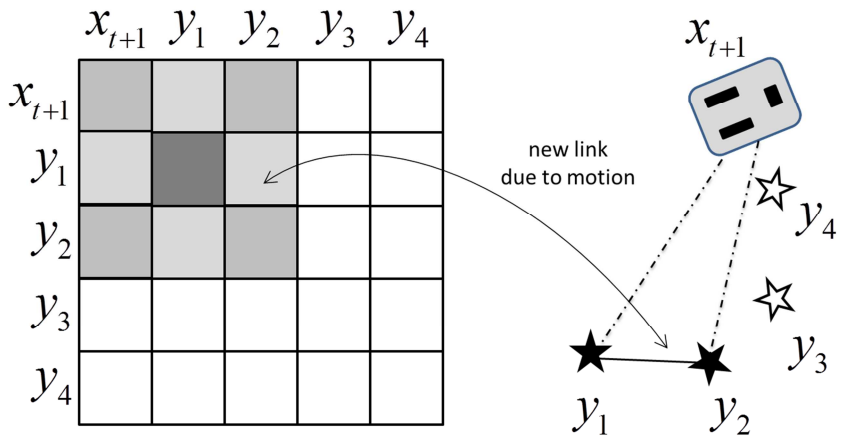


Figure 4.8 : The effect of motion update process.

Lemma 2: If the information matrix is sparse and the Jacobian of the pose change with respect to the absolute robot pose is zero, the motion update requires constant time. This is the case for robots with linear mechanics, and generally, this is not the case since *x*-*y* update depends on the robot orientation. This case is considered in the next lemma.

Lemma 3: If the information matrix is sparse and the mean μ_t is available for the robot pose and all active landmarks, the motion update requires constant time. The mean of the robot pose and all active landmarks cannot be maintained by the information filter, and extracting it via (4.106) is not constant time. For that reason the approximate μ_t is computed by transforming the problem into an optimization problem [56].

4.2.4.3 Sparsification

The final step in SEIFs is the sparsification of the information matrix H_t. The sparseness is achieved by removing links (deactivating) between the robot pose and individual features in the map as shown in Figure 4.9. If this is achieved correctly, it also limits the number of links between pairs of features. Removing a link in the network corresponds to setting the related element in H_t to zero. However, this requires the manipulation of the other links between the robot and other active features, so the resulting network is the approximation of the original one. The details of the sparsification technique is given in [56]. Here, we only give the result of the sparsification. All the landmarks are partitioned to three subsets which are

active landmarks y^+, passive landmarks y^-, and the ones that is sought to deactivated y^0.

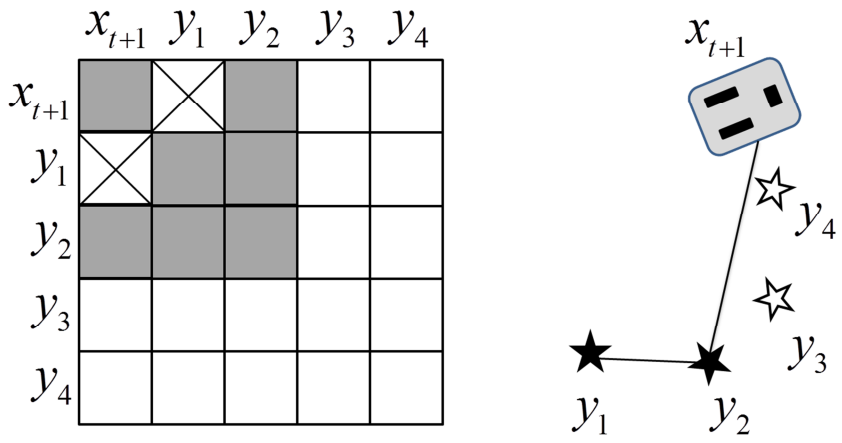


Figure 4.9 : Sparsifications by deactivating the link between the robot and the feature y_1.

$$y = y^+ \cup y^0 \cup y^- \tag{4.110}$$

If $y^+ \cup y^0$ includes all active features, the posterior is factored as

$$\begin{aligned} p(x_t, y \mid z^t, u^t) &= p(x_t, y^+, y^0, y^- \mid z^t, u^t) \\ &= p(x_t \mid y^+, y^0, y^-, z^t, u^t)\, p(y^+, y^0, y^- \mid z^t, u^t) \\ &= p(x_t \mid y^+, y^0, y^- = 0, z^t, u^t)\, p(y^+, y^0, y^- \mid z^t, u^t) \end{aligned} \tag{4.111}$$

In the last step of (4.111), it is exploited the fact that if one know the active features y^+ and y^0, the variable x_t does not depend on the passive features y^-. Therefore, y^- can be set to an arbitrary value like $y^- = 0$ without affecting the conditional posterior over x_t, $p(x_t \mid y^+, y^0, y^-, z^t, u^t)$. To sparsify the information matrix, the posterior is approximated by the following distribution, where the dependence of y^0 is dropped in the first term. This posterior is computed in constant time and the details are given below.

$$\begin{aligned} \tilde{p}(x_t, y \mid z^t, u^t) &= p(x_t, y^+, y^0, y^- = 0, z^t, u^t)\, p(y^+, y^0, y^- \mid z^t, u^t) \\ &= \frac{p(x_t, y^+ \mid y^- = 0, z^t, u^t)}{p(y^+ \mid y^- = 0, z^t, u^t)}\, p(y^+, y^0, y^- \mid z^t, u^t) \end{aligned} \tag{4.112}$$

The matrix $H_t^{'}$ is obtained by extracting the sub-matrix of state variables y^+ and y^0. Here the *S*-matrices are the projection matrices.

$$H_t^{'} = S_{x,y^+,y^0} S_{x,y^+,y^0}^T H_t S_{x,y^+,y^0} S_{x,y^+,y^0}^T \tag{4.113}$$

As a result, the sparsified update equations are obtained as follows.

$$\begin{aligned} \tilde{H}_t &= H_t - H_t^{'} S_{y_0} (S_{y_0}^T H_t^{'} S_{y_0})^{-1} S_{y^0}^T H_t^{'} \\ &\quad + H_t^{'} S_{x,y_0} (S_{x,y_0}^T H_t^{'} S_{x,y_0})^{-1} S_{x,y_0}^T H_t^{'} - H_t S_x (S_x^T H_t S_x)^{-1} S_x^T H_t \\ \tilde{b}_t &= \mu_t^T H_t + \mu_t^T (\tilde{H}_t - H_t) = b_t + \mu_t^T (\tilde{H}_t - H_t) \end{aligned} \tag{4.114}$$

The vehicle model, landmark model, and measurement model given in Section 4.2 are used in the SEIF based SLAM.

SEIF provides an approximate solution to SLAM problem, and the improvements on SEIF are proposed by Eustice et al. First, they introduced exactly delay state filters for view-based systems using scan matching. Second, Walter presented a provably consistent filtering method for imposing sparsity, and called this method Exactly Sparse Extended Information Filters for feature based SLAM [67].

The bottleneck of the information filters including SEIF or ESEIF is that the localization and data association requires the robot pose during the navigation. Therefore, Thrun et al. propose the usage of Markov blanket by computing the conditional distribution instead of solving robot position directly from (4.106), which is an expensive way [57]. In this thesis, a solution to data association using the semantic properties of the planar features is discussed in the next section.

4.2.5 Semantic Data Association with Properties of Planar Features

The data association is the fundamental and critical part of any SLAM method. The conventional data association methods for point features are based on statistical measurement methods. For the planar features, one cannot use the classical approaches since the features are represented by infinite plane parameters. Therefore, other plane parameters, considered as semantic data information, such as convex hull points, number of plane points, and covariance matrix of the plane inlier points, which are not stated in the state vector, can be used to decide correspondence decision. In our experiments, the same criteria proposed in the merging step are also

used in data association as well. Namely, first the two conditions, translations and orientations tests, are investigated. Then if these tests are satisfied, the closeness test is checked. In this approach, there is no need to store any neighboring cell structure or tree.

4.3 Discussions

A planar-feature detection algorithm and its application to outdoor 6D-SLAM is introduced. The conventional plane extraction algorithms are based on two merging tests, which are the normal and the translation error coherency. Although this is valid for indoor structured environments, for outdoor and complex environments, it is added one more constraint to decide whether two planes belong to the same surface. These criteria are also used in the data association problem, which is the bottleneck of the point-feature based SLAM. The method is adapted to local filters EKF, UKF, CKF, RSPKF, and SEIF based SLAM. The main difference between the UKF and CKF is the approximations used to solve the given integrals. While the UKF filters uses unscented transform, CKF uses the cubature transform for solving the integrals. In addition, a new filtering method, RSPKF, based on randomized sigma point sampling is introduced for localization mapping problem. The advantage of the proposed method is that the estimations are unbiased and does not yield the systematic error that is always the case of the classical filtering approaches. Although this approximation takes more computational time, it can be used accurately in SLAM problems without systematic error. Then SEIF are proposed for planar feature based 6D-SLAM. SEIF uses the information form the state space representation and it provides an scalable algoritm for large-scale SLAM by keeping the motion update in a constant time, which is problematic for conventional information filters. Finally, a solution to the data association problem for planar-features by utilizing its semantic properties is introduced. Since the conventional data association methods require the computation of the covariance matrix of the features, their computation time grows as the state vector size increases.

5. EXPERIMENTAL RESULTS

This chapter is dedicated to the experimental results of scan matching and feature based SLAM methods. Totally, three different datasets, which are Kvarntorp Mine, Hannover [73], and Ford [44] datasets, are used in the experiments for several reasons. Kvarntorp Mine dataset are used for the initial results and performance analysis, and since it does not provide the ground truth data, the detailed performance analysis covering the whole dataset that belongs to the Hannover University campus is used. The most of the analysis of the proposed scan matching methods, scan matching based SLAM performances, and feature based SLAM performances are based on the Hannover dataset. In addition to these dataset, the Ford dataset, generated in 2011, is used in the analysis of the proposed feature based scan matching method. Ford dataset provides the laser scan data, camera images, and the ground truth data. This dataset is used to illustrate correspondence of the projection of the extracted features on the camera images.

5.1 Performance Evaluations of Scan Matching Based SLAM Methods

This section is divided into two parts. Firstly, the performance of the scan matching methods is given, and secondly the scan matching based SLAM performances are investigated.

5.1.1 Evaluation of Multi-Layered NDT

In order to test the performance of the ML-NDT, two experimentally datasets, obtained from the Kvarntorp Mine and Hannover University Campus, are used. Both datasets can be obtained from 3D data repository provided by Andreas Nüchter [73]. The Kvarntorp mine is no longer in production, but was once used to mine limestone. The mine consists of around 40 km of tunnels, all in approximately one plane. The resolution of a 3D scan is 361 × 226 data points covering the area of about 180° × 116.3° in front of the robot, and 3D scanning proceed in a drive-scan-and-go fashion. Another experimental dataset, Hannover campus provided by Oliver Wulf

is used [73]. This dataset including 468 3D scans, each with approximately 20,000 data points, was recorded at the Leibniz University Campus. One scan is given by three columns in *x*, *y*, and *z* axes. We use a part of the map and its size is 30m × 60m.

Performance analyses are divided into three parts. In this first part, NDT and ML-NDT are compared in terms of their convergence performance. In addition, the ML-NDT performances with Newton and Levenberg-Marquardt optimizations are discussed. In the second part, the overall performance out of the first 100 scan in the Hannover dataset, and the effect of sampling strategies on the methods are evaluated. Finally, the effect of parameter and starting layer initializations are analyzed, and the results are given.

5.1.1.1 ML-NDT and NDT Performance Comparison

A part of the input scan is used from the dataset obtained from Kvarntorp Mine. Since the point clouds are so close to the each other (almost zero translation and rotation), the input scan is artificially translated and rotated as illustrated in Figure 5.1.

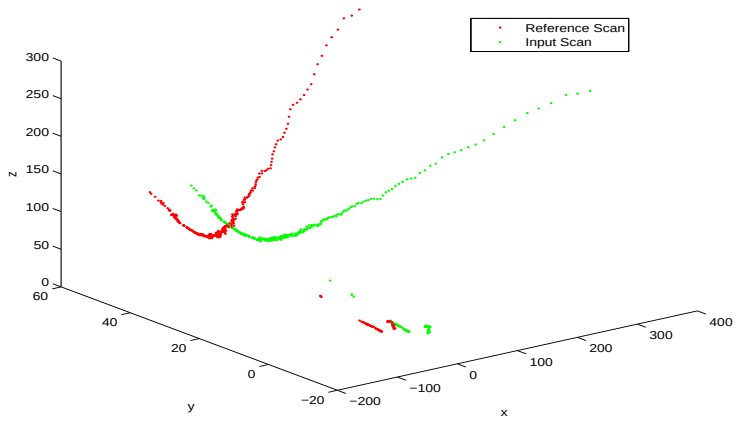


Figure 5.1 : Reference and input scans.

Two 3D point clouds are matched by using ML-NDT and the result is shown in Figure 5.2. The rigid body transformation parameters are successfully estimated even for a long way off movements without any initializations. However, it cannot be said that the algorithm always converges without initializations; therefore, for complex scans the initialization play important role, which is discussed in detail later on.

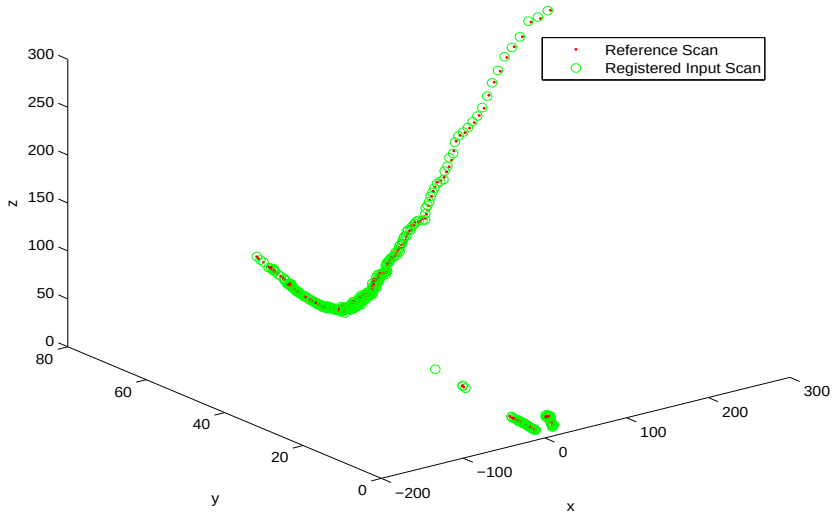


Figure 5.2 : Multi-Layered NDT scan matching result after alignment.

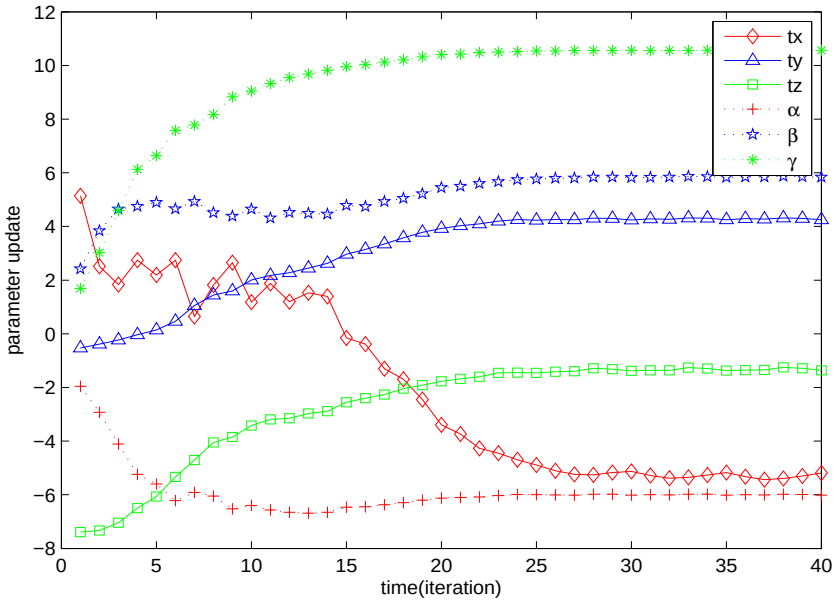


Figure 5.3 : Parameter update of the original NDT.

Parameter update steps in conventional NDT for the same scans are shown in Figure 5.3, where a cell volume is 1/64 times of the volume of point cloud and α, β, and γ are in degrees, others are in metric unit. The cell volume is chosen as 64 times smaller than the volume of point cloud, which corresponds to the second layer representation in ML-NDT.

The related score function variation can be seen in Figure 5.4. To be able to estimate the transformation parameters, they need to be small in original NDT. Otherwise, it does not converge and becomes unstable after a few iterations. To be able to obtain better result in NDT, cell size must be changed by trial and error or initial values

must be close enough to actual ones. Namely, its performance is highly correlated with the quality of odometry or IMU.

The score function becomes steady after 25 iterations for the NDT as seen in Figure 5.4. However, convergence occurs in a few iterations for ML-NDT by using LM method as shown in Figure 5.5.

In addition to the fast convergence property, the transformation parameters can be estimated better than the traditional NDT for long-range variations. While NDT can detect translation up to -10 to 10 units and rotation up to 5 degrees, ML-NDT can detect translations up to 25 units and rotations up to 15 degrees without any initializations. This performance shows the long-range scan matching ability of the ML-NDT with respect to NDT.

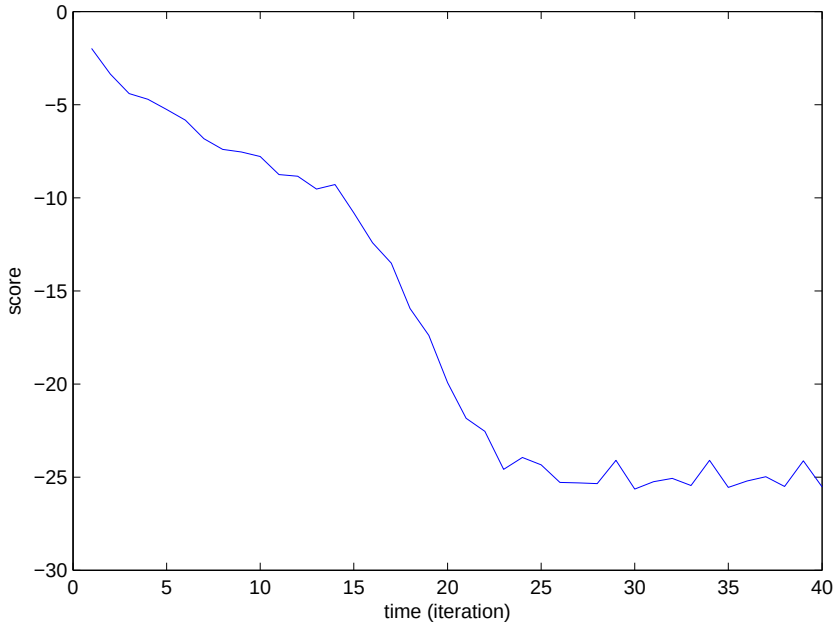


Figure 5.4 : Variation score function of the original NDT versus time (iteration).

During the experiments, it is observed that in some cases, the ML-NDT method can detect up to 45° rotations successfully, which is a significant performance. The normalized score functions of ML-NDT with Newton and ML-NDT with LM are compared in Figure 5.6. ML-NDT with Newton and LM provides very similar result but LM is a little faster, and Newton is more stable after convergence.

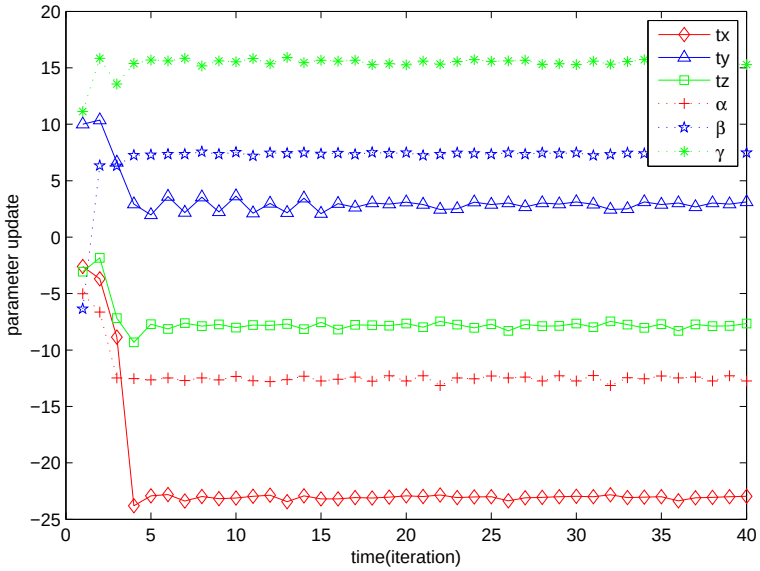


Figure 5.5 : Parameter update of the ML-NDT with LM.

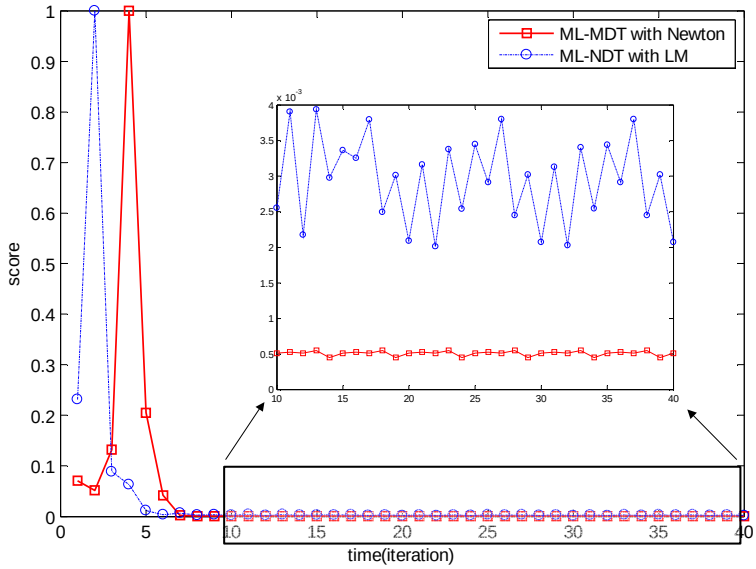


Figure 5.6 : The comparison of ML-NDT with Newton and LM methods.

In the next part, the overall perfomace evaluations and the effect sampling strategies on scan matching methods are investiageted.

5.1.1.2 Overall Performance Evaluations and the Effect of Sampling Strategies

In this part, the performance of ML-NDT and conventional NDT is compared by using the first 100 consecutive scans in the Hannover dataset. Then, the effect of sampling strategies on the algorithm performance is given. For this purpose, this section is divided into two parts, and in the first part, the scan matching methods are compared based on the random sampling. In the second part, they are compared based on the grid based sampling (GBS) strategy, which is presented In Chapter 2.

A. Overall Scan Matching Performance Comparisons

In this part, the relative transformation errors obtained from Odometry, NDT, and ML-NDT are presented. In Hannover 1 dataset, for initial values odometry data is provided in 3D. Namely, two relative translations, t_x and t_y, and one rotation for heading angle θ is available. The aim is to get the relative robot position and orientation in 6D space. Fortunately, the dataset also has the ground truth data in 6D, so it is possible to compare the errors between introduced methods and the ground truth.

In Figure 5.7, the error variation between the odometry and ground truth for three relative transformation parameters is shown.

(a)

(b)

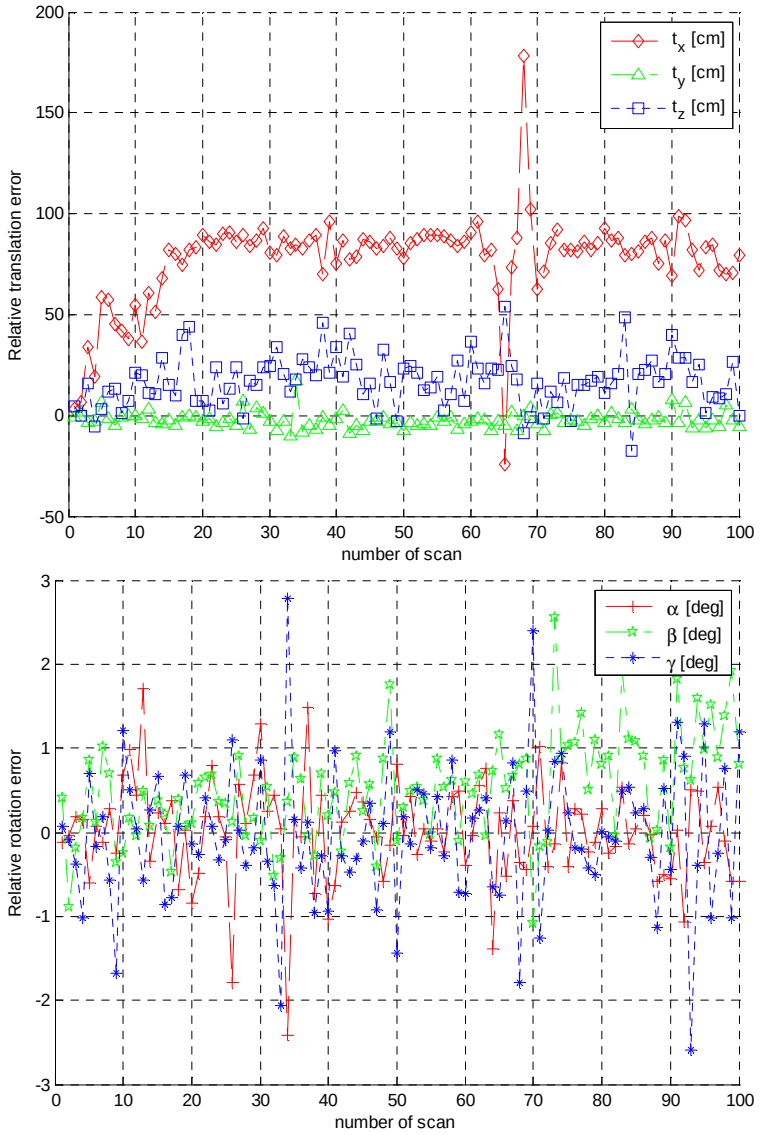


Figure 5.7 : Odometry error variation for the first 100 scans in dataset: (a)Relative translation error variation. (b)Relative rotation error variation.

As seen, after the 20th scan, the error variation is around 80 cm, and it has two pick values at 65th and 68th scans, which makes the scan matching very difficult. The statistical information of the error variation in terms of mean, standard deviation, and maximum values is given in Table 5.1.

Table 5.1 : Error statistics of odometry (N.A: Not Available).

	t_x [cm]	t_y [cm]	t_z [cm]	α [deg]	β [deg]	γ [deg]
Mean	77.52	N.A	16.96	N.A	0.50	N.A
Std.	22.91	N.A	12.678	N.A	0.61541	N.A
Max.	178.61	N.A	53.938	N.A	2.5508	N.A

Table 5.2 : Error statistics of NDT.

	t_x [cm]	t_y [cm]	t_z [cm]	α [deg]	β [deg]	γ [deg]
Mean	19.83	-0.76	1.87	0.09	0.19	0.19
Std.	28.45	2.92	11.68	0. 40	0.72	0.57
Max	125.26	7.59	42.59	1.32	2.70	2.46

Table 5.3 : Error statistics of ML-NDT.

	t_x [cm]	t_y [cm]	t_z [cm]	α [deg]	β [deg]	γ [deg]
Mean	0.97	2.00	-2.25	0.09	0.072	0.19
Std.	17.17	4.60	12.91	0.49	0.42	0.64
Max	90.89	17.68	33.39	1.32	1.01	2.83

As a second analysis, the conventional NDT and ground-truth data is compared. The translational and rotational errors are shown in Figure 5.8. If we compare the errors between NDT and the actual values, it is seen that it obviously provides better results than odometry. The mean of translational errors are reduced up to four times in *x*-axis and almost ten times in *z*-axis. Rotations are also estimated with almost 0.2 degree errors. However, if we consider that the translation errors less than 50 cm are successful, we conclude that the success rate of the conventional NDT is about 90%

percent, which is a similar result of the Magnusson's ICP and NDT comparison study [34]. More statistics on the error variation between ground truth and conventional NDT are given in Table 5.2.

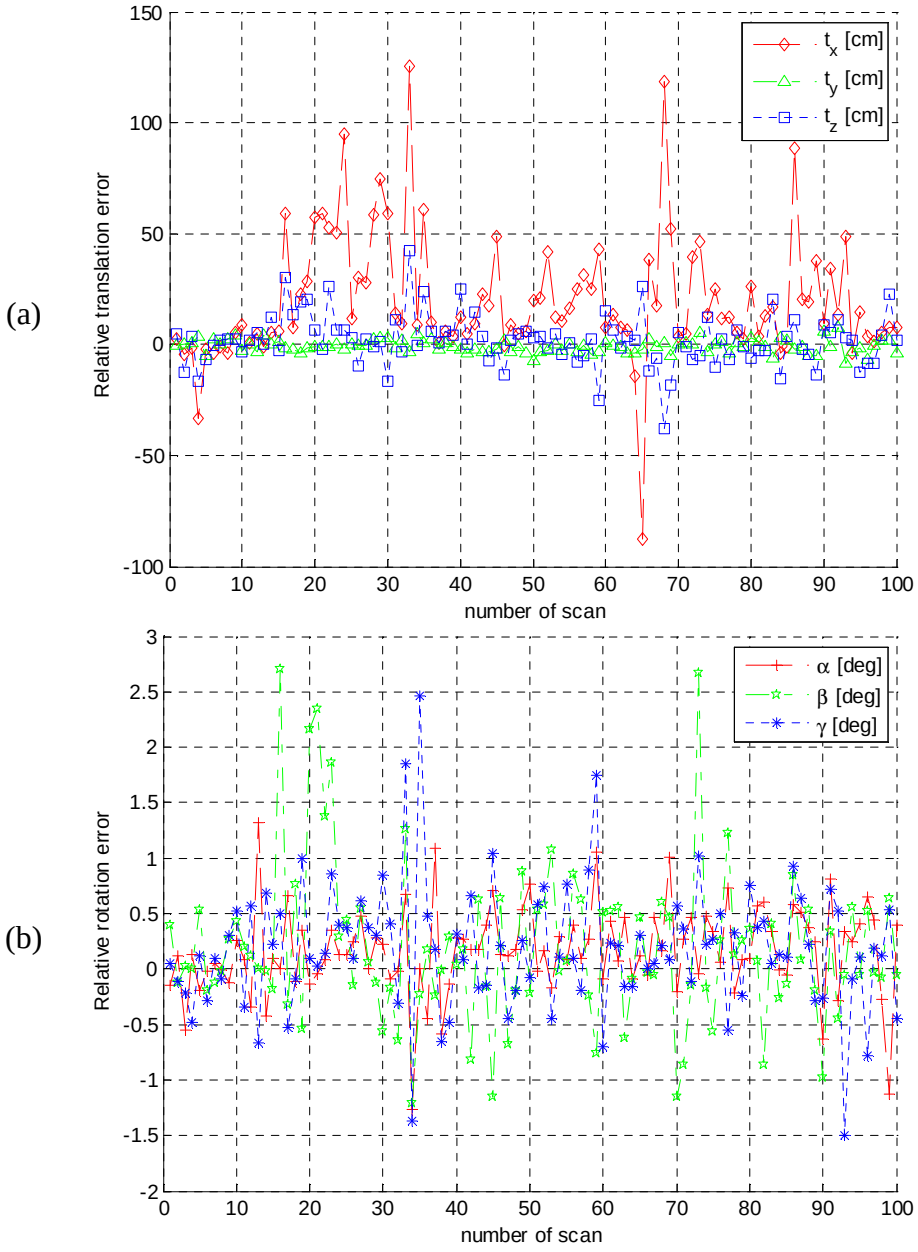


Figure 5.8 : Error variation of NDT: (a)Translation error. (b)Rotation error.

In order to show the performance improvement of the ML-NDT over NDT, the error variation based on ground truth data is given in Figure 5.9. As seen from the figures and the related tables, the results are very satisfactory. The translational error is reduced up to 2 cm and rotational error is reduced up to 0.1 degree angle. The mean of standard deviations of the translational error seems about 15 cm.

(a)

(b)

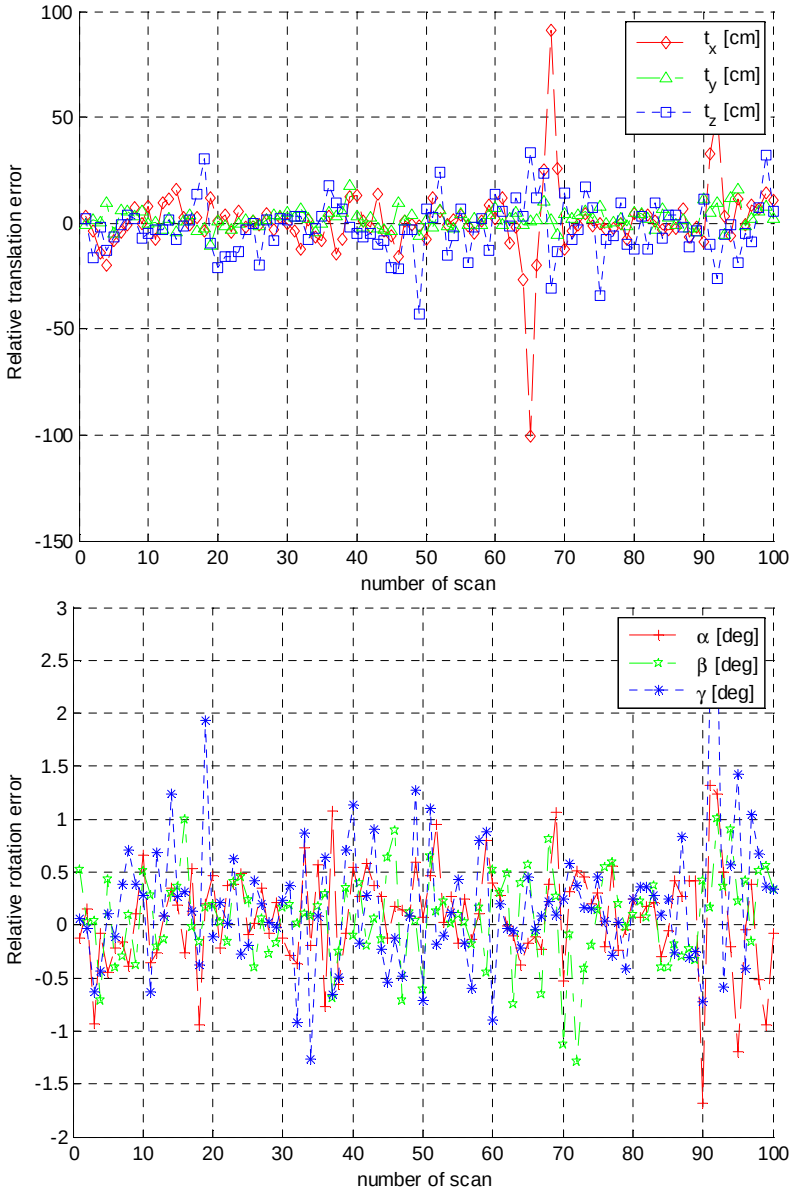
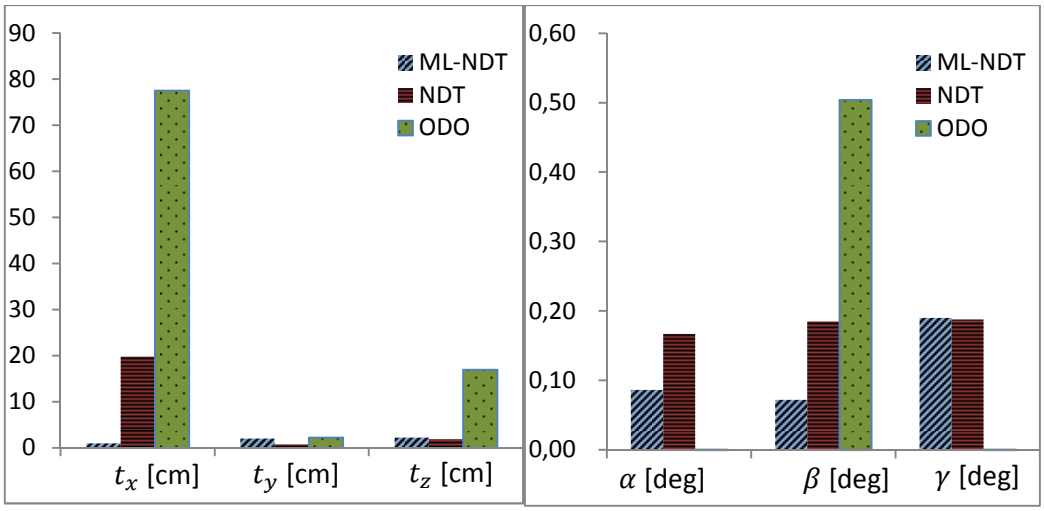


Figure 5.9 : Error variation of ML-NDT: (a)Translation error. (b)Rotation error.

Therefore, the method cannot converge to correct values on these scans. As a result, since only two estimations have more than 50 cm error, we conclude that the success rate of the ML-NDT based scan matching is about 98%. More statistics on the error variation statistics between ground truth and ML-NDT are given in Table 5.3.

If we consider, Magnusson's ICP and NDT comparison study [34], this success rate is almost same as the NDT with tri-linear interpolation. However, tri-linear interpolation almost takes five times computation time, but it is shown that ML-NDT takes less time even than NDT since the algorithm converges faster and the number of iteration almost half of it.



(a) (b)

Figure 5.10 : Error variations of ML-NDT, NDT, and odometry: (a)Translation error. (b)Rotation error.

In Figure 5.10, as an overall performance comparison of three methods, which are odometry, NDT, and ML-NDT, a bar chart showing the average values of the errors of the estimated parameters over the first 100 scans of the dataset is shown.

The NDT and ML-NDT use Newton optimization method in an iterative manner. Therefore, for iterative approaches, one cannot guarantee the computation time. However, by limiting the number of iterations and checking the amount of update on the estimated transformation parameters in two successive iterations, one can compare the computation times of two methods under the same assumptions. This kind of comparison is given in Table 5.4. As it is seen, the computation time of ML-NDT is about a quarter of NDT. Since these experiments are implemented in MATLAB, the computation times can be reduced effectively if they are implemented in a real-time system.

Table 5.4 : Computational comparisons of ML-NDT and NDT.

	Conventional NDT	***ML-NDT***
Mean of comp. time	39.81 seconds	9.43 seconds
Std. of comp. time	9.39 seconds	3.90 seconds
Mean of iterations	25	12.77

B. The Effect of Sampling Strategy on Performance

Sampling method has a great importance on the speed and the robustness of the algorithm. The grid based sampling (GBS) strategy, proposed in Chapter 2, was used in the previous experiments, and here its advantage is emphasized by comparing with conventional sampling methods. In this part, the ML-NDT performance is given for the method of uniform random sampling. However, the point is that, even though we use uniform random sampling, the sampled data is not uniformly distributed since the point cloud distribution is inhomogeneous.

In Figure 5.11, the error variation is given for ML-NDT with random sampling. We see that the success rate, based on 50 cm translational error, is about 94%. One can see more statistics on the average of the absolute values of error variation in Table 5.5, where the failed registrations are not taking into account since their errors, which are obvious in the 14^{th} and 92^{th} scan index, artificially increase the standard deviations.

Table 5.5 : Error statistics of the ML-NDT in case of uniform random sampling.

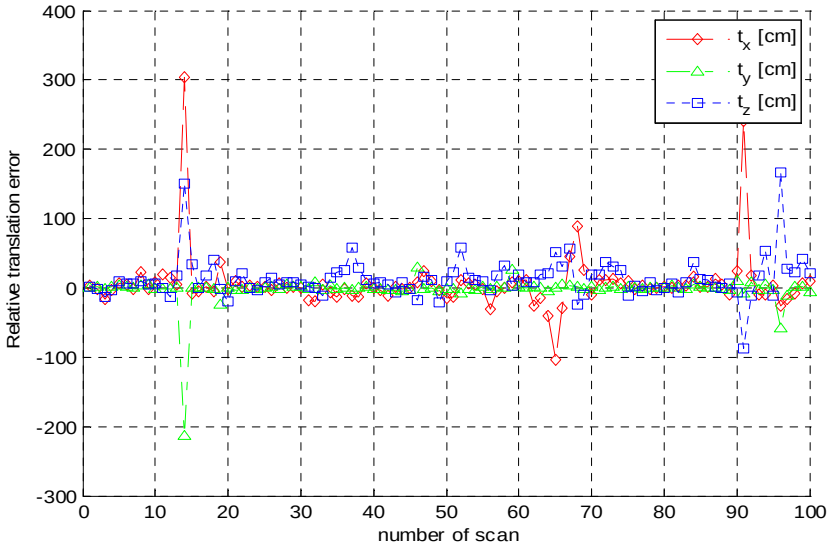
	t_x [cm]	t_y [cm]	t_z [cm]	α [deg]	β [deg]	γ [deg]
Mean	5.90	12.77	1.87	0.09	0.28	0.77
Std.	42.89	22.70	28.75	1.21	2.99	4.62
Max	303.22	28.76	166.66	9.42	28.69	41.30

Table 5.6 : Error statistics of the NDT in case of uniform random sampling.

	t_x [cm]	t_y [cm]	t_z [cm]	α [deg]	β [deg]	γ [deg]
Mean	34.82	2.49	26.39	0.39	1.13	0.55
Std.	24.57	1.72	19.44	0.31	1.07	0.44
Max	132.45	6.15	82.41	1.38	4.98	2.22

Although, there is no figure is given for NDT with random sampling, the error statistics can be seen in Table 5.6 for a comparison.

(a)



(b)

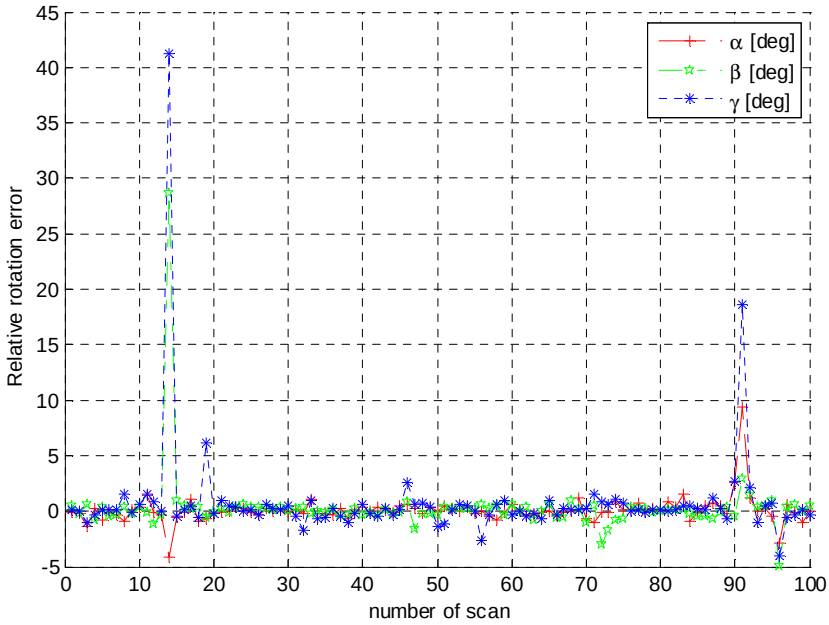


Figure 5.11 : Error variation of ML-NDT with uniform random sampling. (a)Translation error (b) Rotation error.

The norm of translational error of the ML-NDT based method is almost two times better than the NDT based method. The success rate of the NDT with random sampling is about 64%, which is 27% worse than ML-NDT counterpart is. Although there is no separate table is given, the similar results given in Table 5.4 are obtained in terms of the computation times. These results show the robustness, swiftness, and accurateness of the ML-NDT in case of uniform random sampling. In Table 5.7, to compare random sampling and grid based sampling strategies, an emotional

illustration is given based on the statistics on the errors obtained from the experimental results.

Table 5.7 : NDT and ML-NDT comparison based on sampling strategies.

	NDT	***ML-NDT***
Random Sampling		
Grid Based Sampling		

Another important issue of the initialization of relative transformation parameters and the selection of the starting layer is investigated in the next subsection.

5.1.1.3 The Effect of Parameter and Starting Layer Initializations

The performance of the scan matching algorithms is highly depend on the odometry or IMU sensor measurements since this information is used as a starting point in the algorithm. This issue is also known as the initialization problem in scan matching literature. The conventional scan matching algorithms requires a quality odometry data; otherwise, the algorithm tends to diverge if the amount of relative transformation is more than the tolerable amounts. We performed an experiment to be able to analyze the ML-NDT performance based on the knowledge of initial parameters. In the first case, we assume that the odometry data is not available, so the norm of initial parameters are taken as $Z_i=0$. In the second case, the initialization parameters is brought closer to ground truth transformation with 30% and chosen as $Z_i=0.3Z_{gt}$. In the third and fourth case, norm of initial parameters are chosen as $Z_i=0.6Z_{gt}$ and $Z_i=0.9Z_{gt}$, respectively. Another important point is what the starting layer of the algorithm will be.

For this purpose, we combined the analysis of initialization problem and the effect of starting layer on the performance. For the all cases, the algorithm is started from the second, third, and fourth layer and run three times in each layer. The performance analysis of the twelve combinations is given in Table 5.8. The results show that if the relative transformation parameters obtained by the odometry data is very close, representing the fourth case, to the actual transformation parameters, the algorithm can be started from the last (4^{th}) layer due to low fail rate and high speed. The

average translation-error seems about 1 cm better for the starting layer 3; however, the main criteria here are the fail rate and the registration time, which is 60% better. Similarly, in the third case, where the initialization is performed with $Z_i=0.6Z_{gt}$, the algorithm again should be started with the fourth layer due to the same reasons. However, when the initializations are getting worse, namely $Z_i=0.3Z_{gt}$, the algorithm should be started with the third layer due to low fail rate. For the worst case, where no odometry data is available, the algorithm can be started from third or the second layer.

Table 5.8 : The effect of initializations of parameter and starting layer.

Zi/Zg	*Working Layers*	*Fail rate (%)*	*Registration time (s)*	*Error Norm (cm)*
	2,3,4	7	21.65	5.53
0.9	3,4	2	16.05	3.18
	4	**1**	**9.59**	**4.22**
	2,3,4	11	20.88	4.57
0.6	3,4	4	16.63	3.90
	4	**2**	**12.58**	**3.58**
	2,3,4	15	21.13	4.66
0.3	**3,4**	**9**	**21.37**	**3.47**
	4	17	13.70	4.86
	2,3,4	**17**	**21.23**	**6.61**
0	**3,4**	**18**	**17.83**	**3.29**
	4	45	14.00	4.31

In real-time SLAM problem, the ground truth data is not available, so the user might not be able to decide the quality of the odometry data. To solve this problem, we recommend an automatic decision mechanism based on the failure rate. The failure rate is based on the computation of the norm of the error between the estimated and odometry transformation parameters. If this error is greater than a threshold value such as 1 meter for translation and 15 degree for rotation error, then the algorithm is considered as it is failed. Initially, the algorithm is started with the last layer by assuming that the odometry data is satisfactory, and if the failure rate increases up to 5% then the algorithm is started from the third layer. If it still increases up to 20%, the algorithm can be started either from the second or from the third layer. This approach can be easily modified to an adaptive structure to make the scan matching algorithm more robust to the broad range of odometry measurements. This is very important for outdoor and large-scale SLAM problems since the odometry

measurements are obtained from the encoders placed on the vehicle wheels, which might slip easily in some cases. This kind of adaptive structure is planned as a feature study.

5.1.2 Evaluation of Feature Based Scan Matching (FbML-NDT)

This section includes the performance analysis of the feature based scan matching method, FbML-NDT, and the experimental results. Two different datasets, which are the Ford dataset [44] and the Hannover dataset [41], are used for this purpose. This section is mainly divided into two parts. In the first part, an analysis of the feature based scan matching algorithm and the performance of the parameter update step are given. In the second part, the feature extraction method is considered as a special case of sampling strategy and compared to other sampling strategies, which are regular sampling, random sampling, and GBS introduced in this study.

5.1.2.1 Performance of the FbML-NDT

To show the convergence speed of the FbML-NDT algorithm without any initial transformation, the parameter update steps are shown in Figure 5.12.

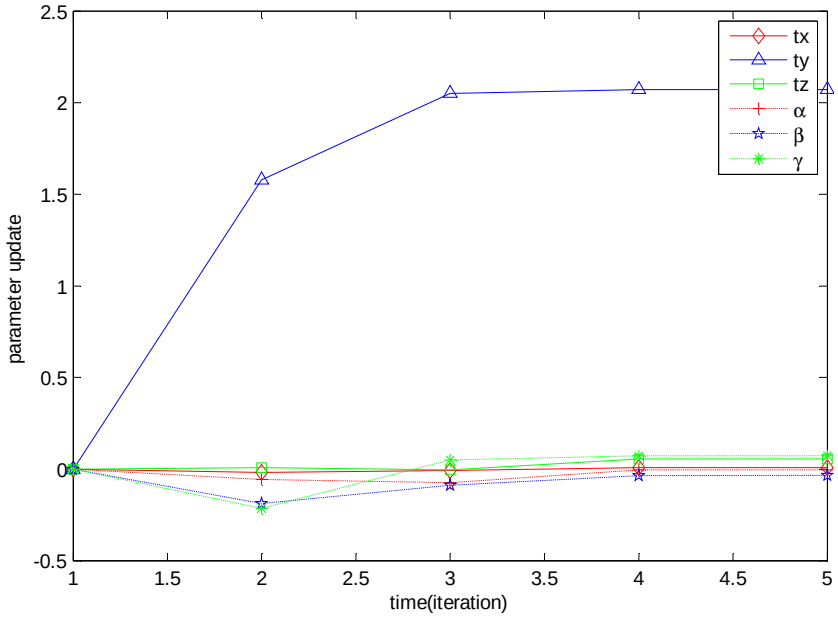


Figure 5.12 : Parameter update steps for Ford dataset [44].

In this registration, the ground truth transformation obtained by GPS and Applanix POS LV sensor is known as 2 meters in *y* direction. During this path, the laser scanner collected five scans, and each scan is obtained approximately with 0.4 meter intervals. These scans are indexed as 1000^{th} - 1004^{th} scans in dataset [44]. It is

observed that the algorithm successfully estimate the correct transformation with a few centimeter error in three iterations. In this registration, no initial transformation parameters are used like odometry and assumed as zero. However, the algorithm may fail if long-range transformations exist. Therefore, an odometry data for initialization improve the system robustness and speed.

Comprehensive results showing the performance of the feature-based scan matching method are given with the comparisons of the sampling strategies in the next subsection.

(a)

(b)

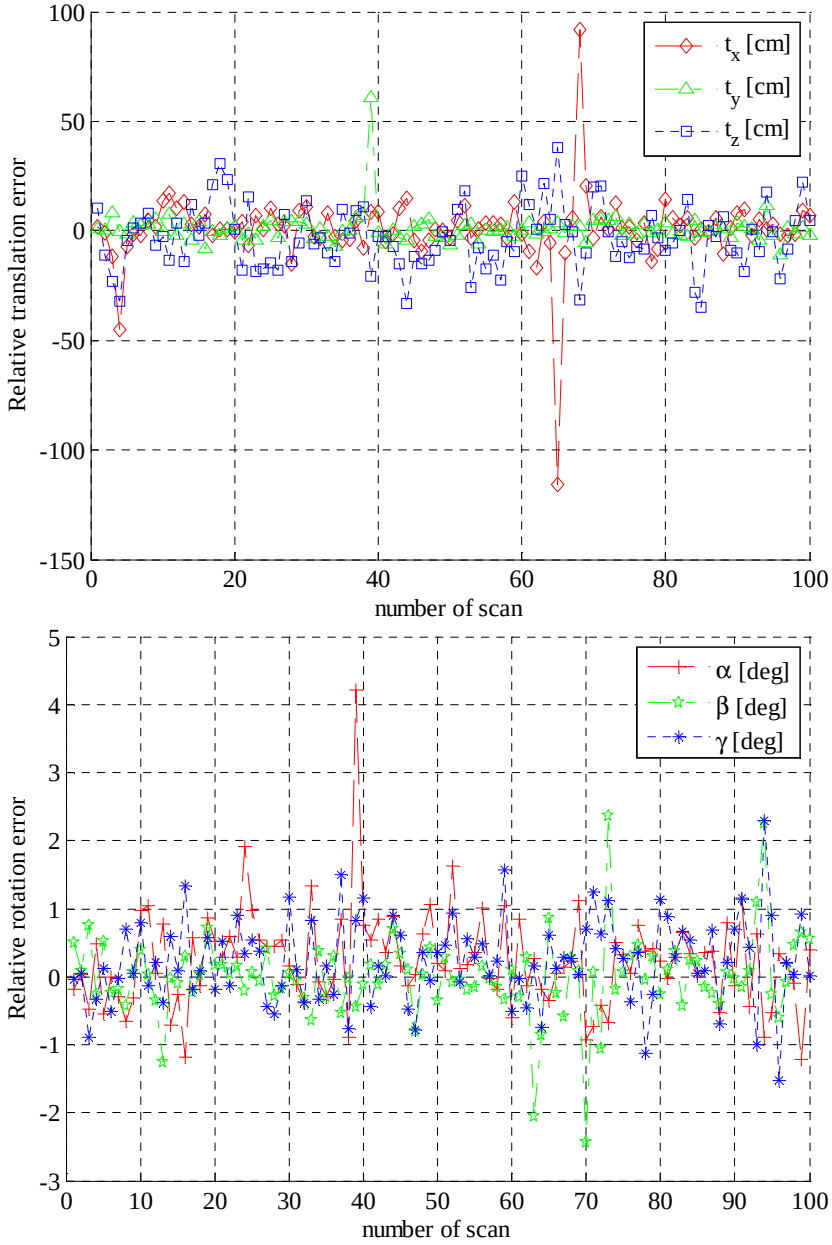


Figure 5.13 : Fb ML-NDT relative translation error variation: (a)Translation error. (b)Rotation error.

5.1.2.2 Comparisons of Sampling Strategies with Feature Extraction

As a further analysis, the feature extraction method is considered as a special sampling method, and the performance of the feature extraction method is compared with the sampling strategies. For this analysis, Hannover dataset is used, and the odometry data given by the dataset is in two dimensions with three parameters, namely, the translation in *x*-*z* axes and rotation (θ) about y-*axis*. The Odometry error variation with respect to ground truth is previously shown in Figure 5.7. In our representations, heading angle corresponds to β, and the all error variations are given with respect to ground truth data provided by the dataset.

The scan matching rotation and translation error variations of the FbML-NDT algorithm is given in Figure 5.13. The odometry data is used as an initial point for all sampling strategies. As it is seen, the mean of the translation and rotation error is very close to zero and the performance is much better than odometry. It is also possible to say that the measurement error can be considered as zero mean Gaussian distribution. The comparisons of all sampling strategies with respect to their translation and rotation angle errors are given in Figure 5.14. As seen from the charts, the performance of the proposed sampling strategies, GBS and FBS, provides satisfactory results. The mean of translation error is less than 2 cm and rotation error is less than 0.3 degree, while the translation error is about 50 cm in odometry and about 7 cm in random sampling case. Furthermore, the error variation can be modeled as normal distribution since it is not biased as in the odometry error. Therefore, Gaussian filters like Kalman and Information filters can be successfully used with the proposed method since they are optimum under the Gaussian noise.

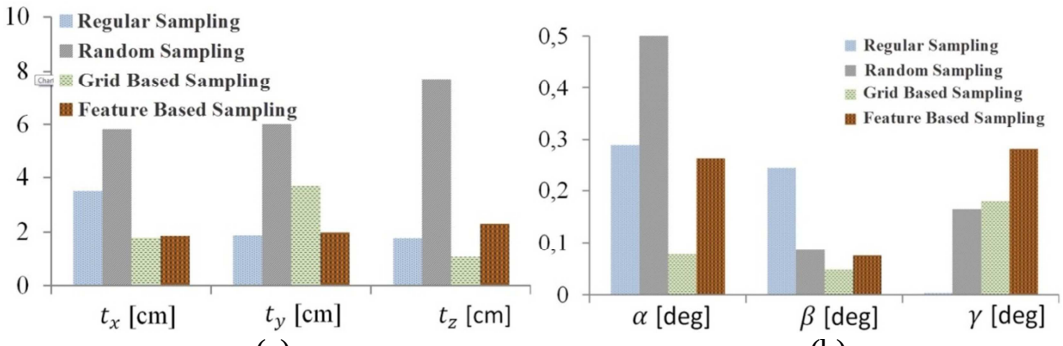


Figure 5.14 : Performance comparisons of all sampling strategies: (a)Average translation errors. (b)Average rotation errors

A more important aspect of the comparisons of the methods is given in Table 5.9. The methods are compared with their other properties, such as the average iteration number, the percentage of matched points, convergence time, and success rate out of 100 scans.

Table 5.9 : Performance comparison of sampling strategies.

	Num. of iteration	***Matched points (%)***	***Matching time (sec)***	***Success Rate (%)***
Regular Sampling	17.27	12.17	44.70	97
Random Sampling	17.24	12.16	43.84	97
Grid Based Sampling	14.29	9.16	7.46	99
Feature Based Sampling	12.67	3.19	2.51	100

In success rate computations, it is assumed that the algorithm is successful if the translation errors are less than 200 cm and rotational errors are less than 10 degree, otherwise it is assumed that the algorithm diverges and is not dependable. As it is seen from table, the success rate is 100%, and convergence time is about 2.5 seconds in MATLAB. These show that the proposed feature extraction method is more robust and faster than the other sampling strategies. By considering the total processing time, it is seen that the feature extraction process requires approximately 1 seconds and grid based sampling requires about 0.1 seconds.

5.1.3 Evaluation of Scan Matching based SLAM

FbML-NDT and odometry localization performances without using any filtering method out of 100 scans are shown in Figure 5.15 (a). In Figure 5.15 (b), the error norm of the robot position with respect to ground truth is shown. While the odometry based navigation is very poor due to biased error, the FbML-NDT based navigation is fairly good. At the end of the path, the absolute error is more than 60 meters for odometry based localization. On the other hand, this error is about 7 meters if FbML-NDT is used.

(a)

(b)

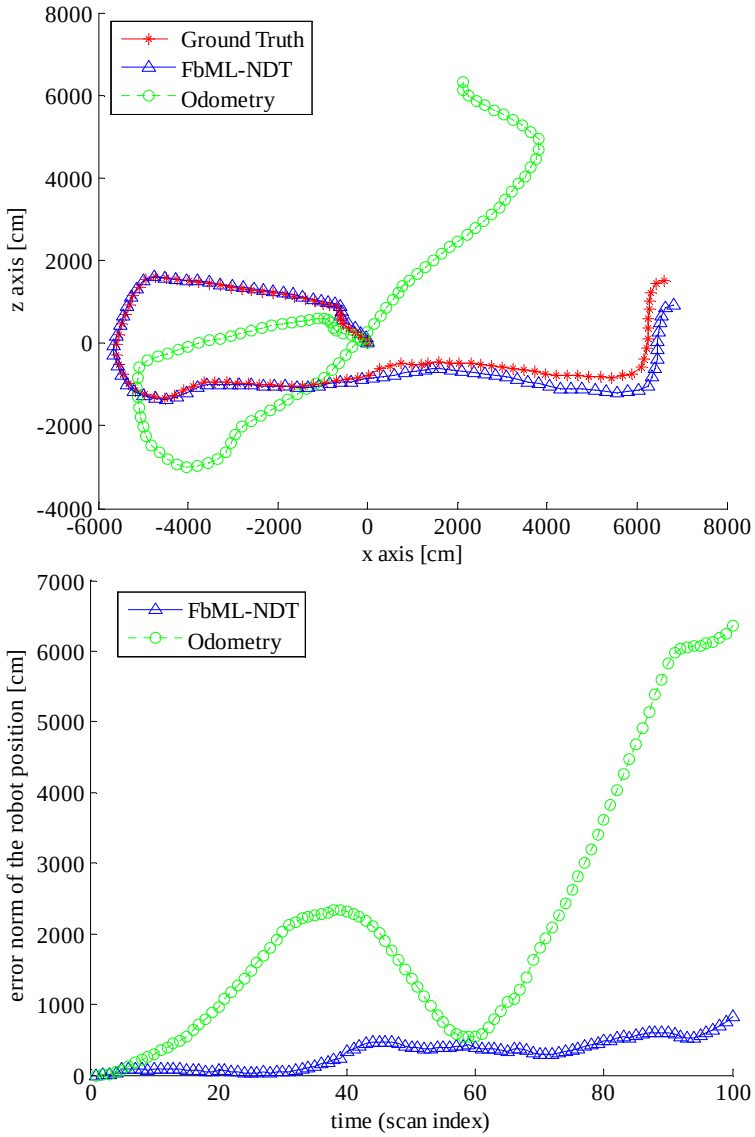


Figure 5.15 : (a) FbML-NDT robot localization performance: (b)The norm of the robot position *(x,y,z)* error.

The above SLAM method does not utilize a filtering method, so it generates cumulative error. However, this error can be reduced if it is combined with a filtering method like EKF. The details of the EKF and scan matching based SLAM are explained in Chapter 3. In next subsection, EKF and ML-NDT based SLAM implementation is simulated in 2D.

5.1.3.1 EKF and ML-NDT Based SLAM Performance in 2D Space

The method is implemented in 2D simulation environment. Figure 5.16 shows the 2D rectangular map with a circle inside. This map is generated artificially for simulation purpose. The robot is shown as triangular object and its ground truth path

is given. The aim of this simulation is to track the robot position and orientation by using ML-NDT with EKF.

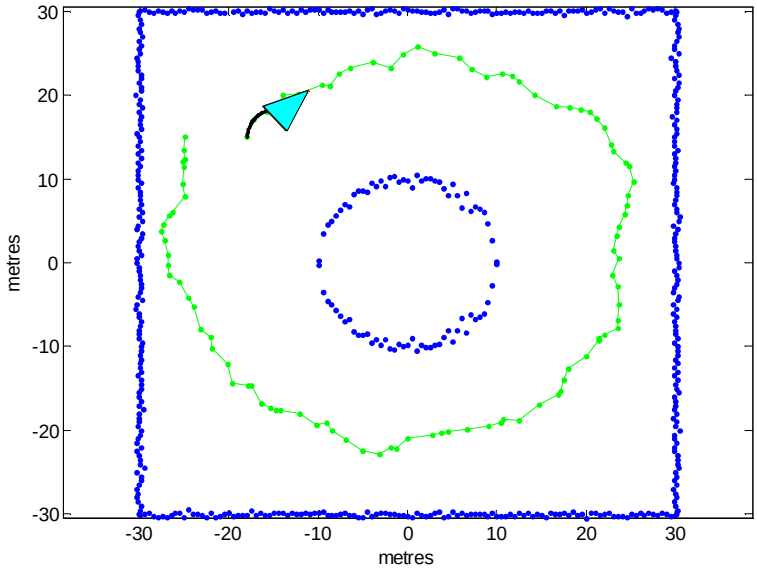


Figure 5.16 : 2D simulation environment**.**

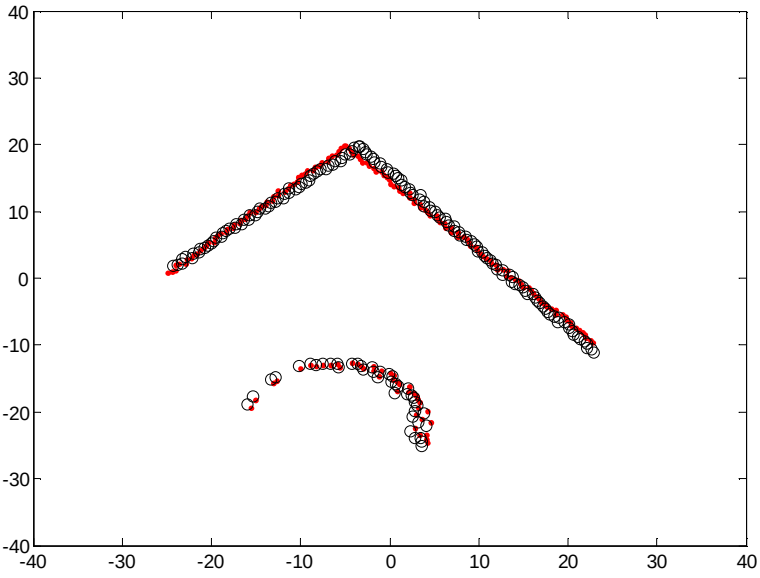


Figure 5.17 : Scan matching result at robot position given in Figure 5.16.

The measurement unit is based on the scan matching; thus, a scan matching result is shown in Figure 5.17 at the robot position given in Figure 5.16. In this simulation, it is assumed that the mobile robot can scan 360 degrees of its surrounding. Figure 5.18 shows the estimated path after the robot travels two times the environment. The lag between the real and actual position is seen from the figure.

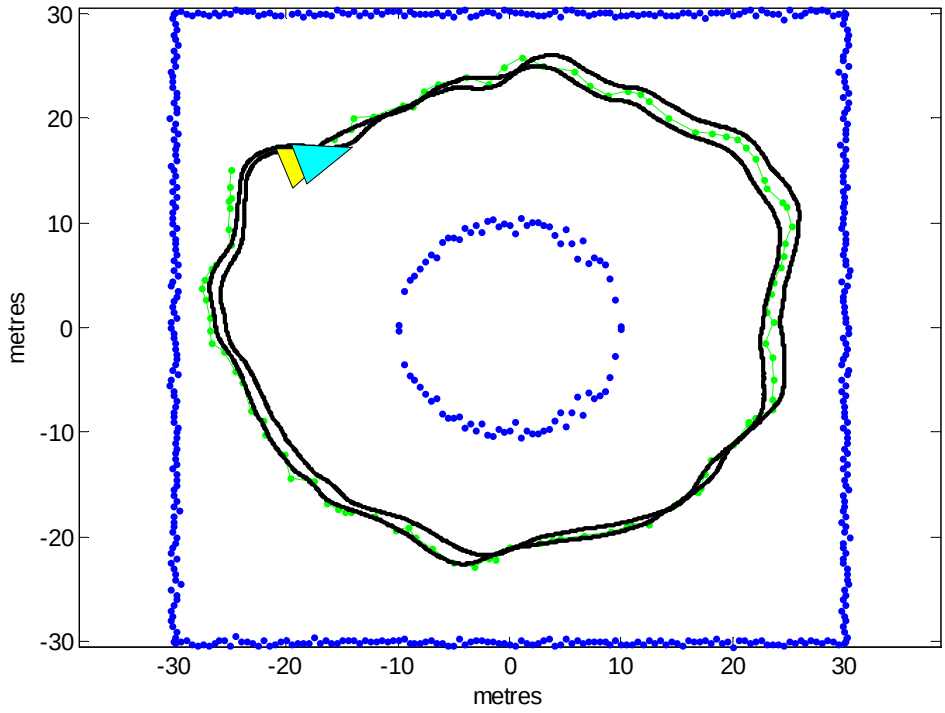


Figure 5.18 : Robot path after two loop tracking.

5.1.3.2 EKF and ML-NDT Based SLAM Performance in 3D Space

The experimental results of EKF and ML-NDT based SLAM for the first 60 scan of the Hannover Campus Dataset is given in this section. 2D Odometry (blue spots) and 3D proposed method navigation with its exaggerated covariance plots at specific positions, which are 15^{th}, 30^{th}, 45^{th}, and 60^{th} positions.

The overall system performance show that 2D odometry position data seems as the projection of the 3D proposed position as shown in Figure 5.20. In case of disagreement of ML-NDT and odometry positions, an alarm flag is raised and the odometry position is accepted as correct. We encounter this kind of case only once throughout the experiment. The performance comparisons of the NDT and ML-NDT scan matching methods have already been given in Section 5.1.1.1. For this reason, the comparasions of the NDT and ML-NDT based SLAM performances are not investigated. However, more comprehensive performance analyses of the NDT and ML-NDT based SLAM methods are left as a feature study.

Performance evaluations of the feauture based SLAM are explained in the next section, which is the second main contribution of this thesis.

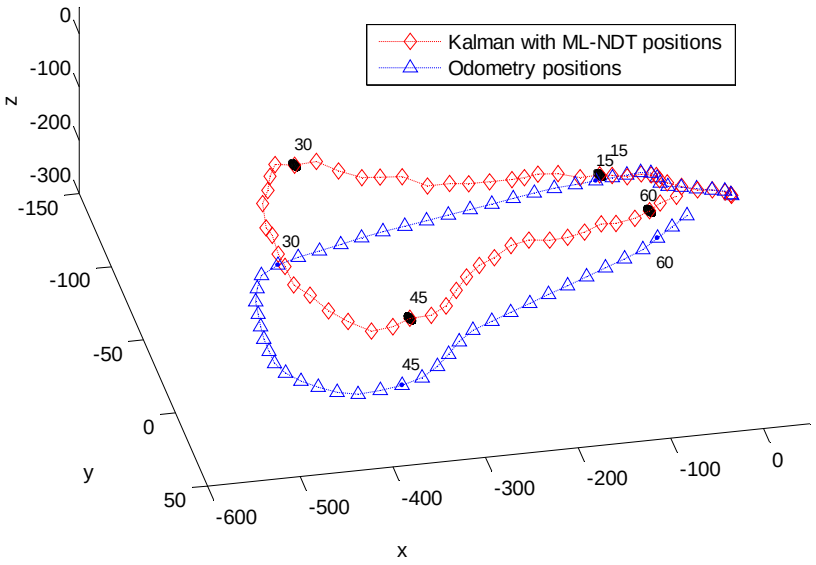


Figure 5.19 : ML-NDT and KF based SLAM performance.

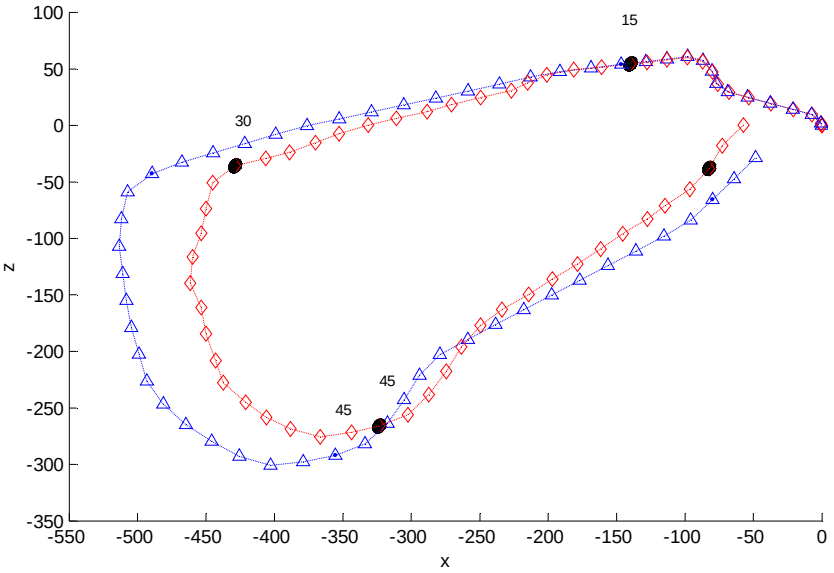


Figure 5.20 : ML-NDT and KF based SLAM performance.

5.2 Performance Evaluation of Feature Based SLAM

We will show the performance of the algorithm for an experimental dataset. The dataset obtained from the Hannover Campus [31] is used and has the ground truth data. The dataset has totally 468 3D scans having approximately 20.000 points.

As described in Section 3, firstly, the point cloud is discretized, and the plane parameters are found by using the RANSAC algorithm. If the number of inlier points is more than the threshold value, the planes are projected to their main principal axes, and their convex hull points are computed and re-transformed to 3D space. The

extracted plane segments are shown in Figure 5.21. Finite planes are obtained by founding their convex hull. This point cloud belongs to the first scan of the dataset. Then the horizontal planes are discarded to eliminate the ground effect since the planes belong to the ground does not satisfy the distinctiveness property.

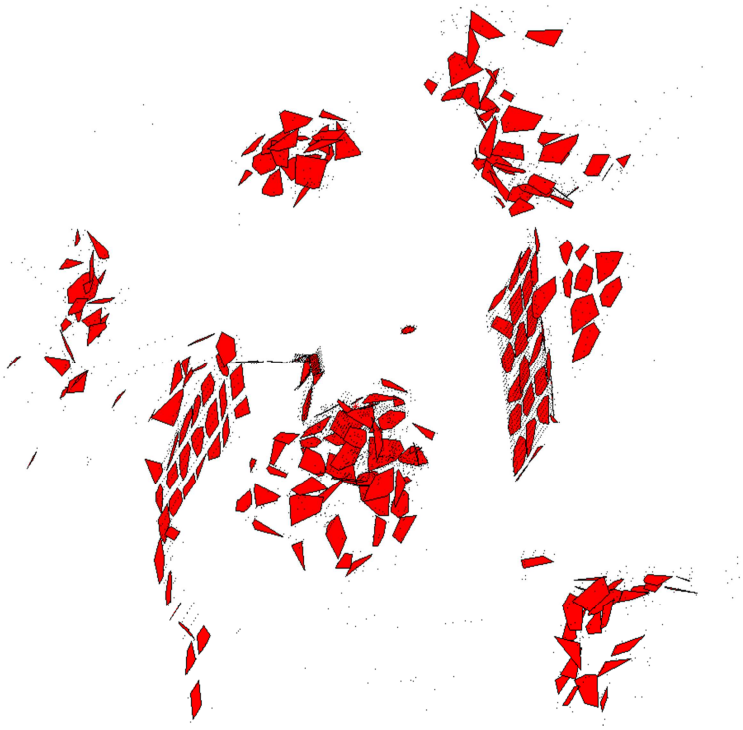


Figure 5.21 : Extracted plane segments.

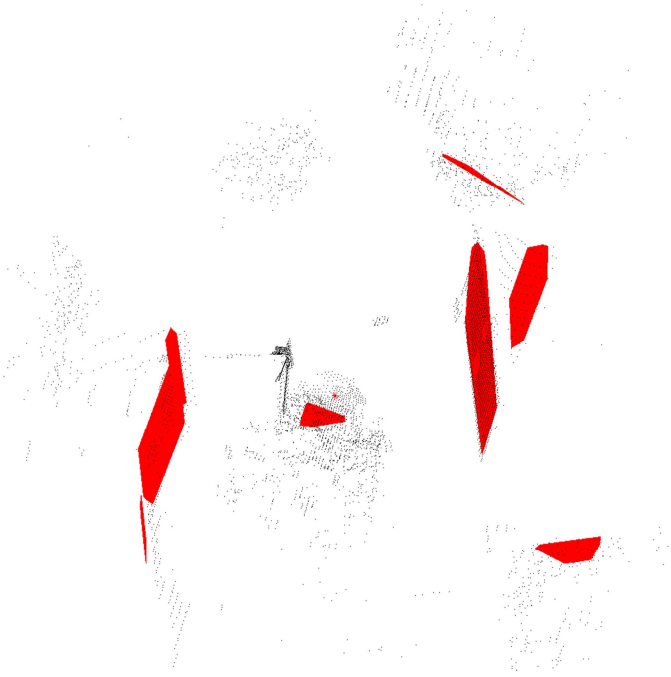


Figure 5.22 : Merged plane segments of the first scan. Detected eight features.

The plane segments are merged by using the proposed merging algorithm and shown in Figure 5.22. There are totally eight features extracted from the point cloud.

With the same conditions, the features are extracted from the second scan and shown in Figure 5.23 for a comparison. There is a trivial robot motion between these two scans. The data association procedure is applied, and the available three features are corresponded successfully. In this data association, since the movement is trivial, there is no need an odometry data for an alignment.

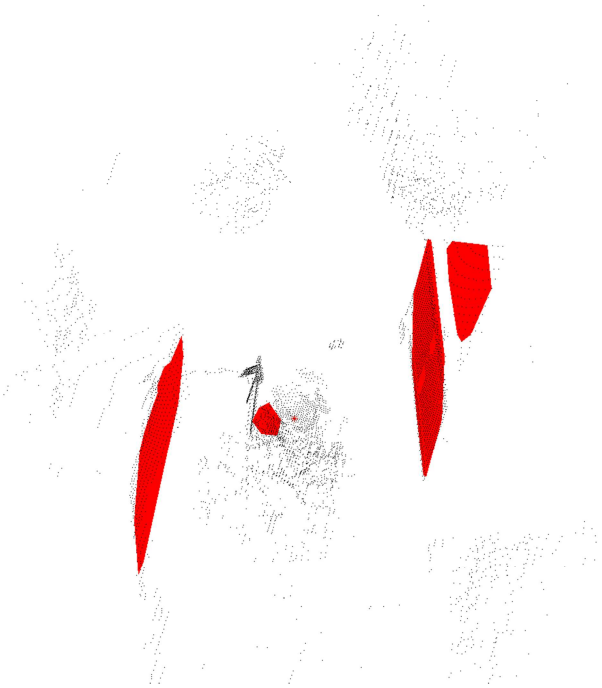


Figure 5.23 : Merged plane segments of the second scan.

However, this is not always the case. Sometimes the robot movement is very extreme as between the 3rd and 4th scan. The rigid body transformation parameters between these two scans are given as follows;

Rotation (Euler) angles (x,y,z) : 0.18, -51.5, 0.37 [degree].

Translations (x,y,z): -35.4, -3.37, 93.75 [cm].

Robot turns around about 51.5 degrees, and this cannot be dealt without an initial guess of the robot motion. Since we have the ground truth data, and by using this information, it is obtained the Table 5.10 for the corresponded features. In real-time implementations, since the ground truth data does not exist, the predicted robot pose information is used. The merging conditions should be relaxed in data association as well**.**

Table 5.10 : Data association performance between the 3rd and 4th scans.

$feat.$	c_1 (deg)	c_2 (cm)	c_3 (cm)
1	2.5	19.5	10
2	1.05	4.4	7
3	0.61	15	5
4	1.21	3.42	8

The criteria proposed in the merging phase are also used in the data association of the 3rd and the 4th scan features. One of the features of both scan is corresponded as shown in Figure 5.24. Finally, the map with plane features and the robot path based on ground truth data is show in Figure 5.25.

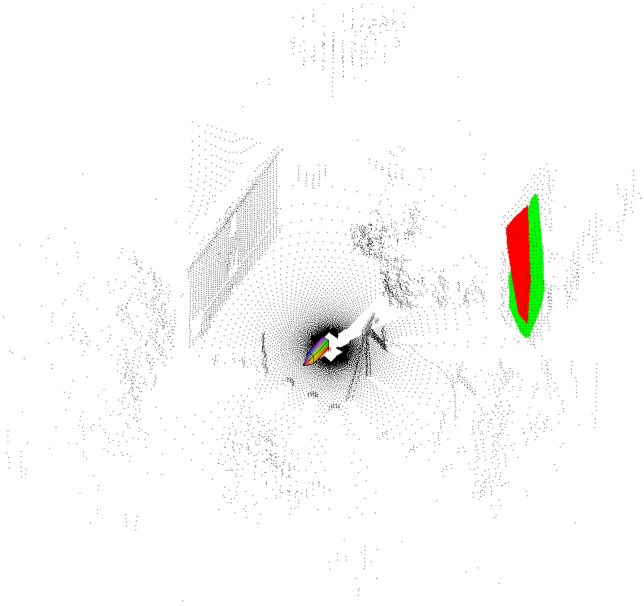


Figure 5.24 : Correspondence of two features.

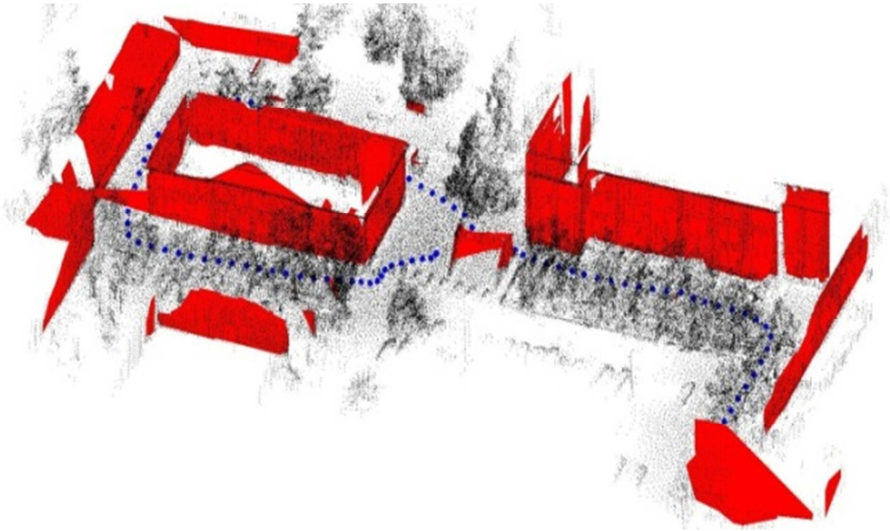


Figure 5.25 : Map and the extracted plane features. The robot poses (blue dots).

5.2.1 Performance Evaluations of Feature Based SLAM Methods

In this section, for the performance analysis of the planar-feature based SLAM method for Hannover dataset generated by Oliver Wulf is used. The single 3D scans as well as initial pose estimates are given in 3D, as x_v, y_v, and ϕ_v. The ground truth pose data is available in 6D as $x_v = [x_{v_p} \quad x_{v_o}]$, where $x_{v_p} = [x, y, z]$ and $\mathbf{x}_{v_o} = [\alpha, \beta, \gamma]$. The vehicle motion, observation, and augmentation models are given.

Vehicle Model The vehicle model function *f* is given by (3.1), and it can be disclosed explicitly as in (5.2) for the odometry data having the relative rigid body transformation parameters.

$$\mathbf{x}_{v_k} = f(\mathbf{x}_{v_{k-1}}, \mathbf{u}_{v_{k-1}}) + w_{k-1} = F\mathbf{x}_{v_{k-1}} + \mathbf{u}_{v_{k-1}} + w_{k-1} \tag{5.1}$$

The odometry data is provided by the relations of $\mathbf{u} = [\delta x, \delta y, \delta z, \delta\alpha, \delta\beta, \delta\gamma]$. The vehicle state vector is represented by $\mathbf{x}_v = [\mathbf{x}_{v_p} \quad \mathbf{x}_{v_o}]$ where $\mathbf{x}_{v_p} = [x, y, z]$ and $\mathbf{x}_{v_o} = [\alpha, \beta, \gamma]$ denote the robot position and the orientation, respectively.

$$\begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ z_{k-1} \end{bmatrix} + Rot(x_{v_{o,k}}) \begin{bmatrix} \delta x_{k-1} \\ \delta y_{k-1} \\ \delta z_{k-1} \end{bmatrix}$$
$$\begin{bmatrix} \alpha_k \\ \beta_k \\ \gamma_k \end{bmatrix} = \begin{bmatrix} \alpha_{k-1} \\ \beta_{k-1} \\ \gamma_{k-1} \end{bmatrix} + \begin{bmatrix} \delta\alpha_{k-1} \\ \delta\beta_{k-1} \\ \delta\gamma_{k-1} \end{bmatrix} \tag{5.2}$$

where *Rot* matrix represent the three successive rotations defined by the Euler angles in *x*, *y*, and *z* axes as

$$Rot(X_o) = \begin{bmatrix} \cos\alpha\cos\gamma & -\cos\beta\sin\gamma & \sin\beta \\ \cos\alpha\sin\gamma + \sin\alpha\sin\beta\cos\gamma & \cos\alpha\cos\gamma - \sin\alpha\sin\beta\sin\gamma & -\sin\alpha\cos\beta \\ \sin\alpha\sin\gamma - \cos\alpha\sin\beta\cos\gamma & \cos\alpha\sin\beta\sin\gamma + \sin\alpha\cos\gamma & \cos\alpha\cos\beta \end{bmatrix} \tag{5.3}$$

Observation Model The observation model is based on the feature extraction method proposed in Chapter 4. The plane features are used as landmarks and are encoded in the state vector with their infinite plane representations. The observation model *h* is given by

$$z_k = h(\mathbf{x}, \mathbf{u}_{k-1}) + v_{k-1}$$
$$z_k = \begin{bmatrix} n_{F_k}^L \\ d_{F_k}^L \end{bmatrix} = \begin{bmatrix} Rot^T(x_{v_{o,k}}) n_{F_k}^W \\ d_{F_k}^W + (n_{F_k}^W)^T x_{v_{p,k}} \end{bmatrix} + v_{k-1} \quad \textbf{(5.4)}$$

where $n_{F_k}^L$ is the plane normal vector represented in the local (L) frame, and $d_{F_k}^L$ is the plane minimum distance to the robot location $x_{v_{p,k}}$ provided by the feature extraction method. The reader is referred to [62] for more information about the feature extraction method. Based on the robot orientation $x_{v_{o,k}}$ and location $x_{v_{p,k}}$ the plane patch parameters are transformed to the world (*W*) frame for state augmentation.

State Augmentation Model The state augmentation model is given by

$$n_F^W = Rot(x_{v_o}) n_F^L$$
$$d_F^W = d_F^L - (n_F^W)^T x_{v_p} \quad \textbf{(5.5)}$$

The infinite plane representations in local and world frame are shown in Figure 5.26. For the data association purpose, the other plane properties such as center of gravity of the planes G_F^L, the covariance matrix C_F^L of the plane points and convex hull points $\Delta^L{}_{XYZ,F}$ are also transferred to the world frame by using the estimated robot position.

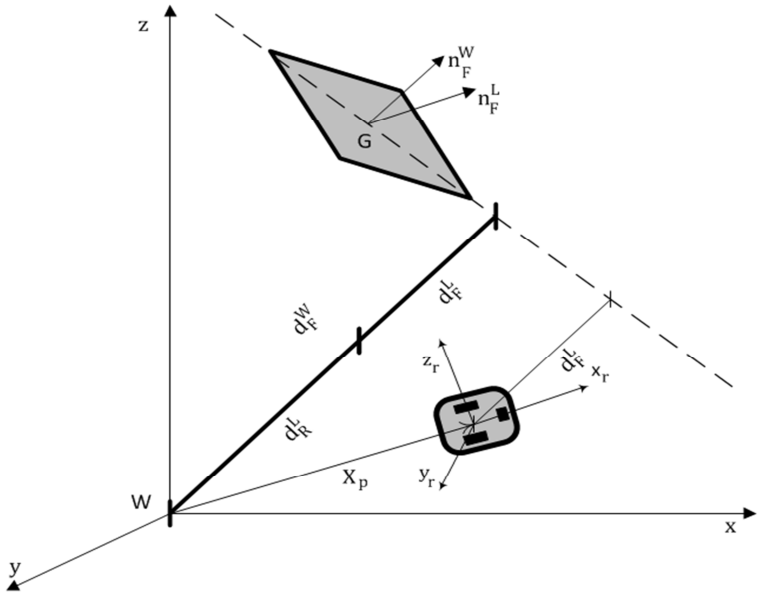


Figure 5.26 : Infinite Plane representation in local and world frame.

$$
\begin{aligned}
C_F^W &= Rot(x_{v_o}) C_F^L \\
G_F^W &= Rot(x_o) G_F^L + x_{v_p} \\
\Delta^W{}_{XYZ,F} &= Rot(x_{v_o}) \Delta^L{}_{XYZ,F} + x_{v_p}
\end{aligned}
\tag{5.6}
$$

In the next section, the performance of two conventional filtering methods, EKF and UKF based SLAM, are given.

5.2.1.1 Comparison of EKF and UKF Based SLAM

The filter consistency is measured by the error norm with respect to the ground truth data. The comparisons of the EKF, UKF, and Odometry results are shown in Figure 5.27. Altogether, we observe that both SLAM methods provide much better results than Odometry data. The process noise covariance matrix $Q = diag(\sigma_{x_v}, \sigma_{y_v}, \sigma_{\phi_v})$ is set by $\sigma_{x_v} = \sigma_{y_v} = 50$ cm and $\sigma_{\phi_v} = 1°$.

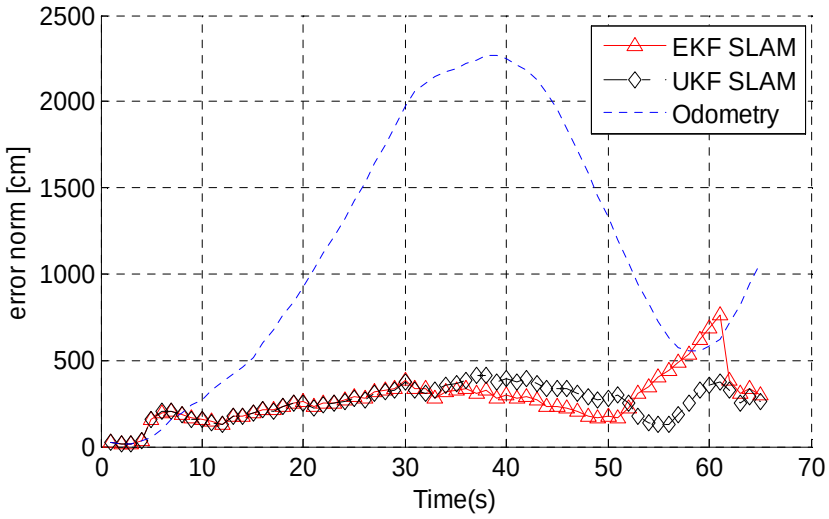


Figure 5.27 : Comparisons of error norms of the vehicle position

The covariance matrix measurement noise $R = diag(\sigma_{n_x}, \sigma_{n_y}, \sigma_{n_z}, \sigma_{n_d})$ is set by $\sigma_{n_d} = 10$ cm and $\sigma_{n_x} = \sigma_{n_y} = \sigma_{n_z} = 0.001$. These assumptions are related to the noise type and its level of the process and measurement errors. The maximum error norm is seen in the 40th scan about 22 m for Odometry, in the 60th scan about 8 m for EKF SLAM, and in the 38th scan about 4 m for UKF SLAM.

The comparisons are extended, and the EKF, UKF, and CKF based SLAM performances are given in the next part.

5.2.1.2 Comparison of EKF, UKF, and CKF Based SLAM

In this section, particularly, the CKF based SLAM performance is given and it is compared to EKF and UKF methods. The vehicle and observation models are the same with previous step.

In Figure 5.28, the absolute error norm of the robot position in three dimensions is given under high level Gaussian, where the process noise covariance matrix $Q = diag(\sigma_x^2, \varepsilon^2, \sigma_z^2, \varepsilon^2, \sigma_\theta^2, \varepsilon^2)$ is set by $\sigma_x = \sigma_y = 50$ cm and $\sigma_\theta = 1°$. The state covariance matrix is initialized by 10^{-6} times identity matrix. As seen, the CKF and UKF based SLAM methods provide almost the same performances. EKF based SLAM method yields inconsistent state estimations in some cases as in the 50th time stamp. The error increases up to 2 meters between two scan. In the 60th scan, the error norm and the size of their uncertainty ellipsoids decrease notably since the landmarks observed at the beginning of the loop is re-observed.

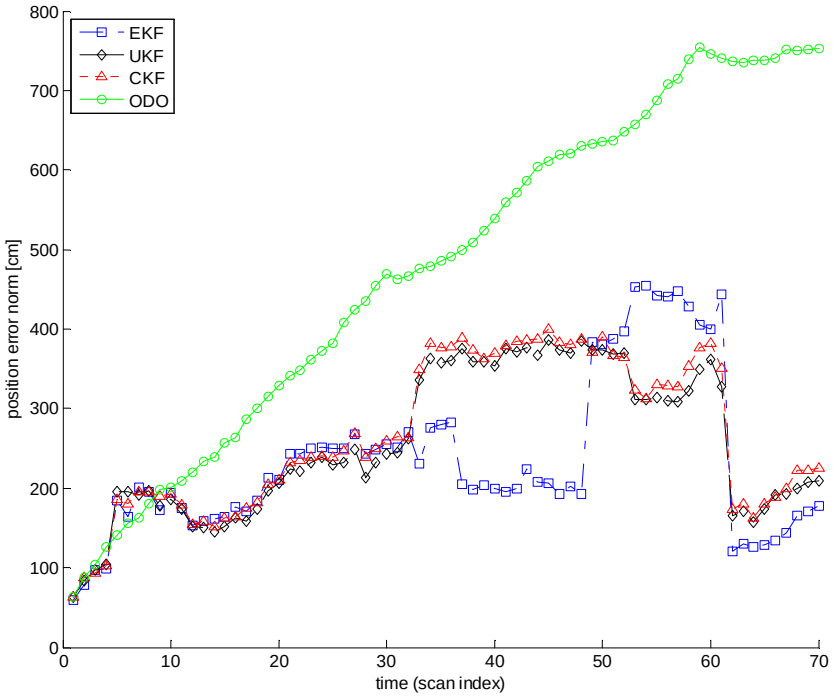


Figure 5.28 : Comparisons of error norms of the vehicle position.

Actual odometry based absolute position error norm is shown in Figure 5.29. As seen, the odometry error norm reaches to 23 meters, and UKF and CKF SLAM performances are superior to the EKF. The error norm never reaches 3 meters for UKF and CKF SLAM and 5 meters for EKF SLAM.

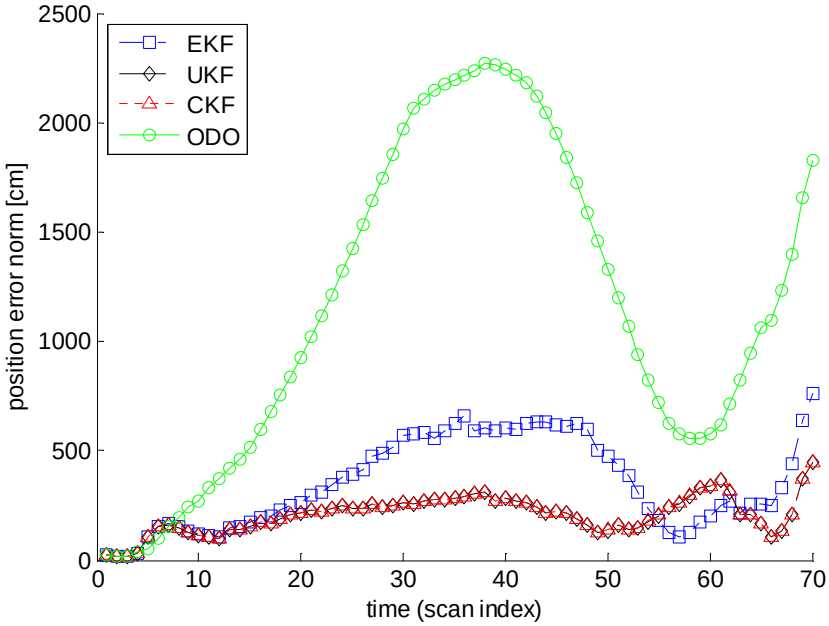


Figure 5.29 : Actual odometry based position error norm.

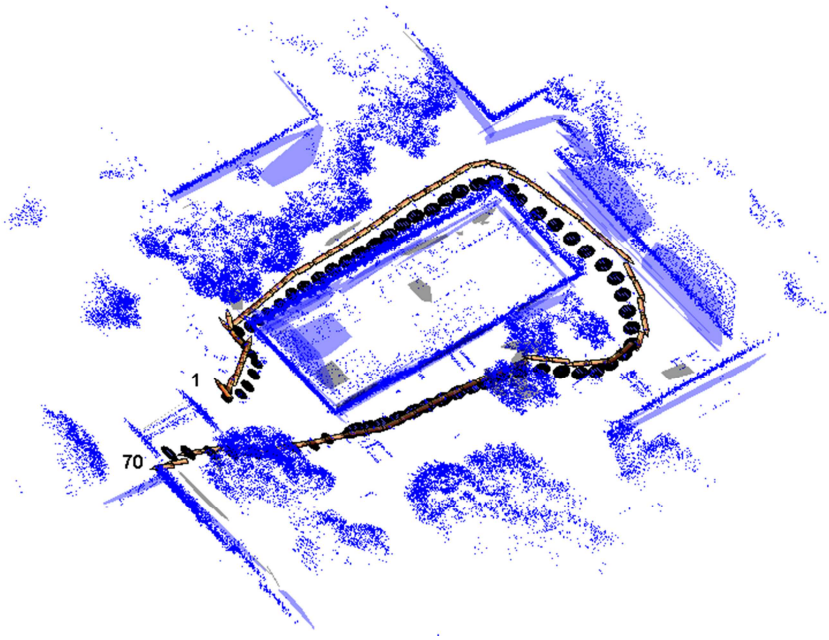


Figure 5.30 : Estimated planar map of the environment. The estimated robot positions with their uncertainty ellipsoids and ground truth path.

In Figure 5.30, the planar map constructed form the SLAM and the actual robot path is given. In addition, the robot 3D position uncertainty ellipsoids with the mean are shown on the map. Here the point cloud is registered based on the ground truth as the reference.

Finally, UKF and CKF performances are compared based on the initial value of the state covariance matrix. When the size is chosen 10^3 times greater than the above

initialization, UKF generates inaccurate results as shown in Figure 5.31. If this is increased more than this value, the UKF loses the positive definiteness and the state covariance matrix cannot be invertible.

Furthermore, the map is increased more than two times, and the vehicle motion model is modeled by the relative transformation, which is a better assumption than the linear model. The vehicle model function given by *f* is disclosed in (5.2) for the odometry data having the relative rigid body transformation parameters.

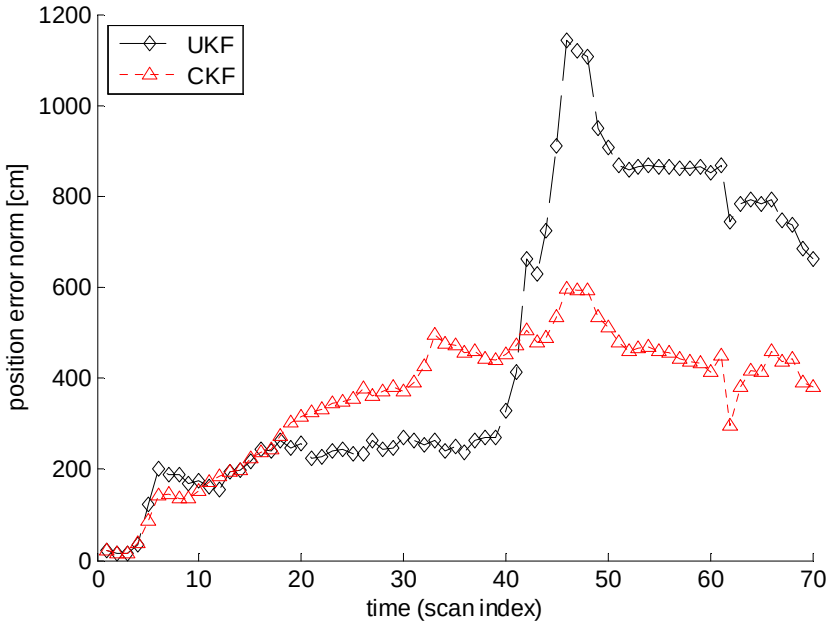


Figure 5.31 : Initialization problem of UKF.

As seen from the Figure 5.32, the odometry data **u** provided by the data set is neither Gaussian nor the zero mean. Therefore, we used the simulated odometry data produced from the ground truth by adding normal random noise with zero mean and 20 cm standard deviation to the relative translation vector and 1° error to relative rotation vector.

The covariance matrix measurement noise $R = diag(\sigma_{n_x}, \sigma_{n_y}, \sigma_{n_z}, \sigma_{n_d})$ is set by $\sigma_{n_d} = 10$ cm and $\sigma_{n_x} = \sigma_{n_y} = \sigma_{n_z} = 0.001$. These assumptions are constant, and they are related to the noise type and its level of the process and measurement errors. In Figure 5.33, the maximum error norm is seen at the 200th scan about 22 m for Odometry, and between the 70th and 190th scan about 5 m for EKF and UKF SLAM out of 211 scans, covering the 160m x 60m area. On the other hand, EKF based

SLAM occasionally becomes inconsistent and its instant pose might have large errors than UKF although they have very similar performance.

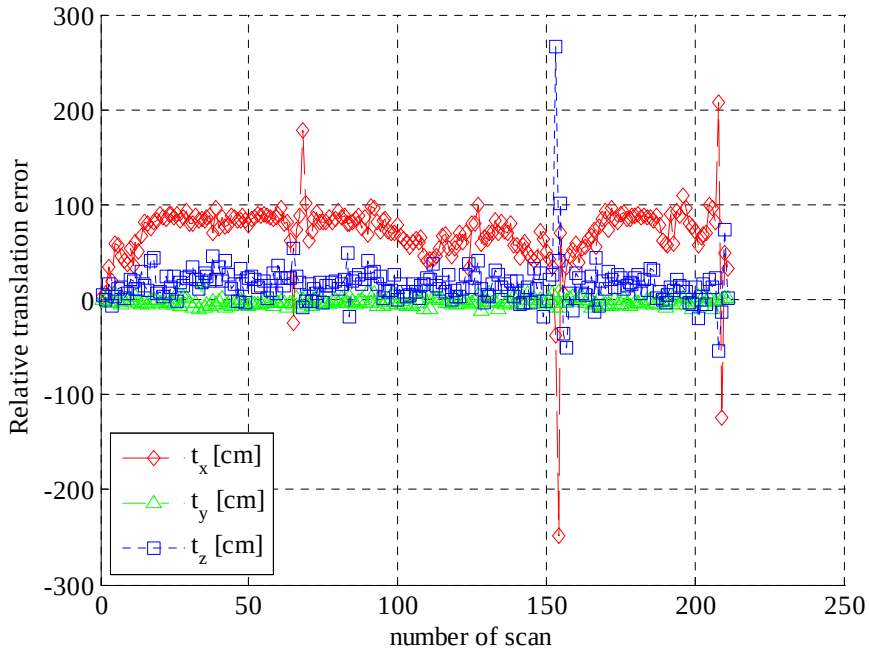


Figure 5.32 : Odometry relative translation error.

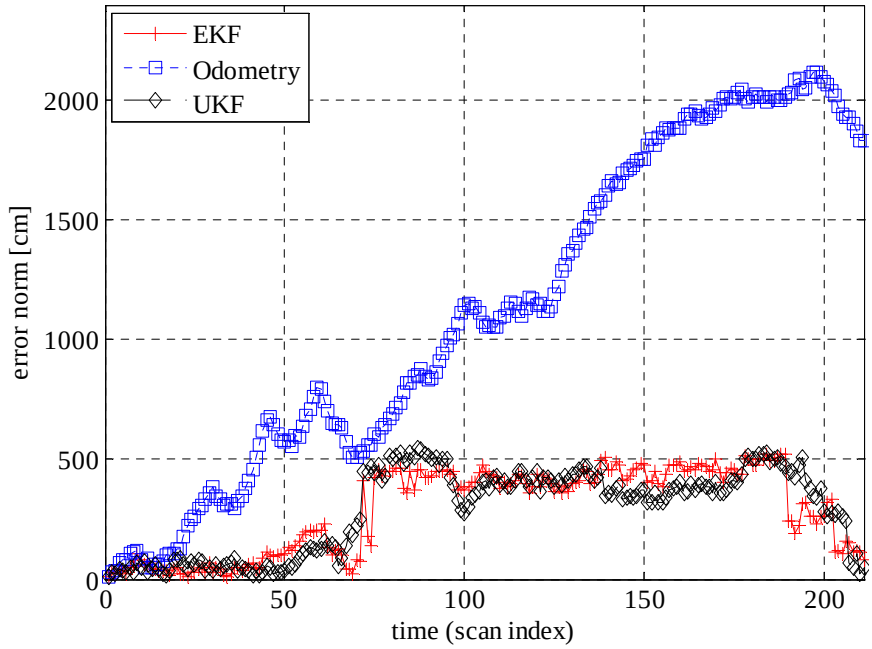


Figure 5.33 : Comparisons of EKF and UKF error norms of the vehicle position.

In Figure 5.34, the estimated robot trajectory, odometry, and ground truth data is shown. The results show that UKF and EKF based localization provides less than 50 cm absolute error at the end of a long loop while zero-mean odometry based localization error is about 20 m.

5.2.1.3 Performance Evaluations of the RSPKF Based SLAM

Randomized Sigma Point Kalman Filters (RSPKFs) are special form of UKF filters using stochastic integration rule (SIR) in their approximations. In this part, the SLAM performance of the RSPKF is compared to conventional Gaussian filters. This section is divided into two part: First, the simulation results in 2D are given. Second, the 3D experiemental results are presented. Before the simulation results, the vehicle motion, observation, and augmentation models are explianed in the next section.

A. Simulation Results in 2D

In this section, an artificial environment containing landmark position in 2D is generated and the robot waypoints are defined. The aim of the SLAM algorithm is to estimate the landmark positions and last robot pose information using the range and bearing observation model.

Vehicle Model The explicit vehicle model is given by

$$\begin{aligned} x_{v_k} &= x_{v_{k-1}} + Vdtcos(\gamma_{k-1} + \phi_{v_{k-1}}) + w_{x_{k-1}} \\ y_{v_k} &= y_{v_{k-1}} + Vdtsin(\gamma_{k-1} + \phi_{v_{k-1}}) + w_{y_{k-1}} \\ \phi_{v_k} &= \phi_{v_{k-1}} + Vdtsin(\gamma_{k-1}) / L + w_{\phi_{k-1}} \end{aligned} \quad \textbf{(5.7)}$$

where V and γ are the control input representing the constant velocity and steering angle with zero mean Gaussian noise *w*, respectively.

Observation Model The range and bearing observation model is given by

$$\begin{aligned} r &= \sqrt{(x_m - x_{v_k})^2 + (y_m - y_{v_k})^2} + \nu_{r_k} \\ b &= \tan^{-1}\left(\frac{y_m - y_{v_k}}{x_m - x_{v_k}}\right) - \phi_{v_k} + \nu_{b_k} \end{aligned} \quad \textbf{(5.8)}$$

where r is the range and b represents the bearing measurements, and the measurements are with zero mean Gaussian noise ν .

State Augmentation Model The state augmentation model $\mathbf{x}_{m_k}$ is given by

$$\begin{aligned} x_{m_k} &= x_{v_k} + r\cos(\phi_{v_k} + b_k) \\ y_{m_k} &= y_{v_k} + r\sin(\phi_{v_k} + b_k) \end{aligned} \quad \textbf{(5.9)}$$

(a)

(b)

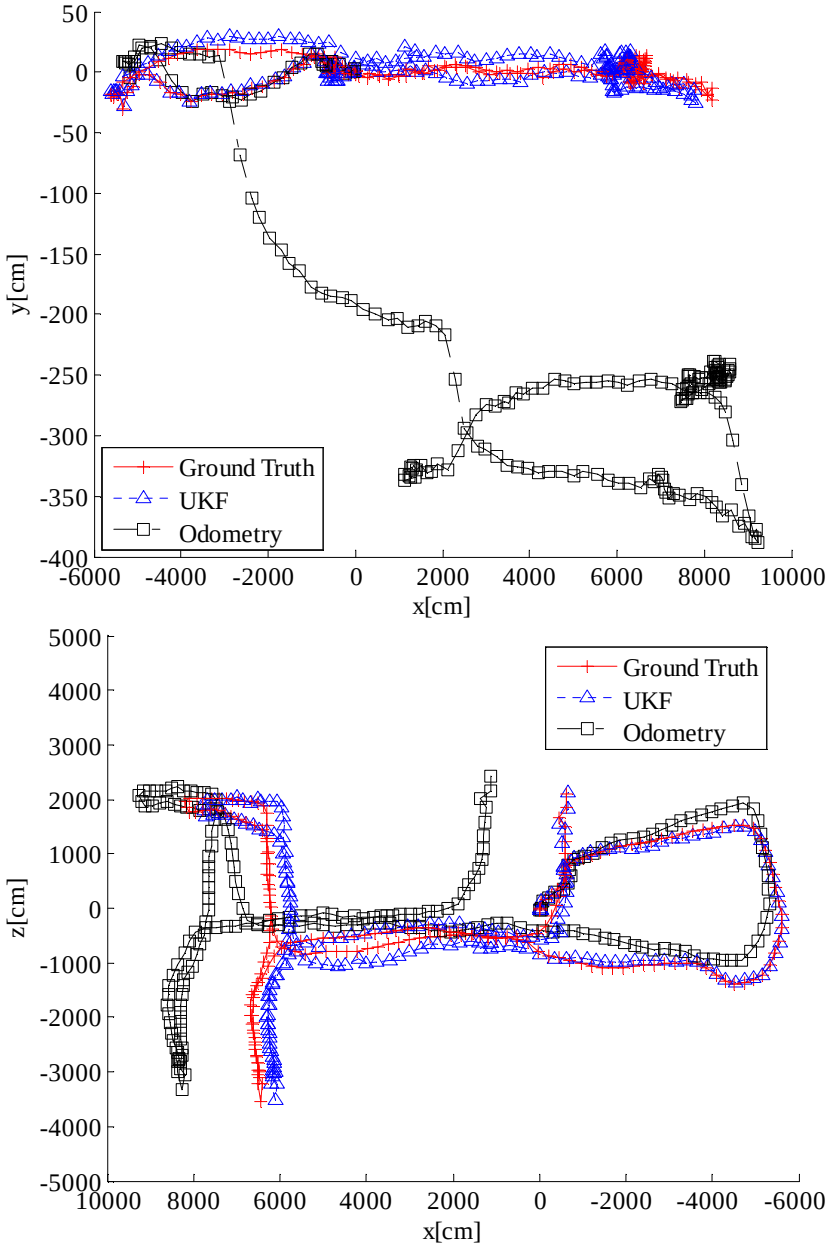


Figure 5.34 : Estimated robot trajectory: (a)*xy* view. (b)xz view.

The Monte Carlo simulations are carried out and the average position and orientation error norms for UKF and RSPKF SLAM methods are shown in Figure 5.35. The control noise is 1 meter in speed and 1 degree in steering angle. Similarly, the measurement noise in range and bearing is assumed as 1 meter and 1 degree, respectively. Therefore, the process covariance matrix Q=R=*diag*(1, π/180). The vehicle speed is taken as 2 m/s and time interval between two control signals is set by 0.05 seconds. The time interval between the two observations is assumed as 2.5 seconds. The feature map and the estimated paths based on the filtering methods are shown in Figure 5.36.

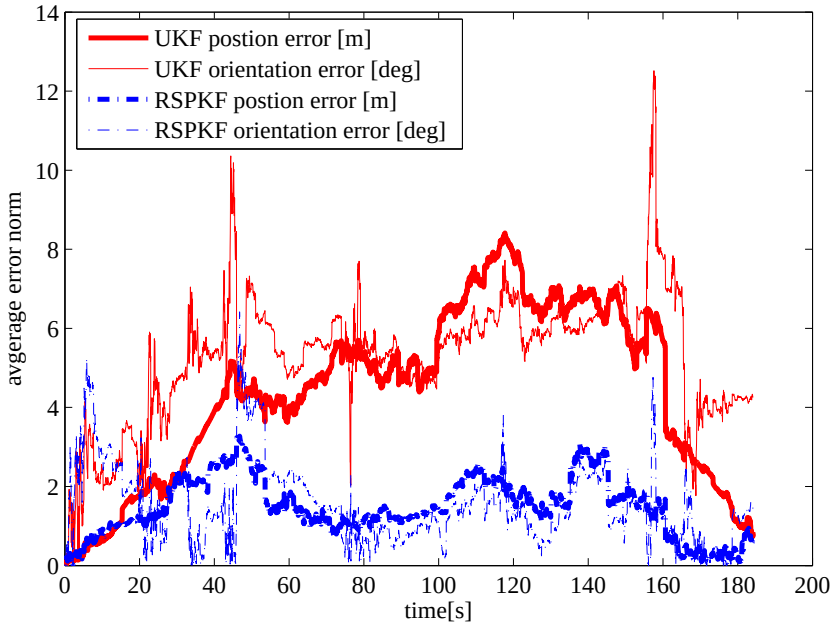


Figure 5.35 : Average position error norm.

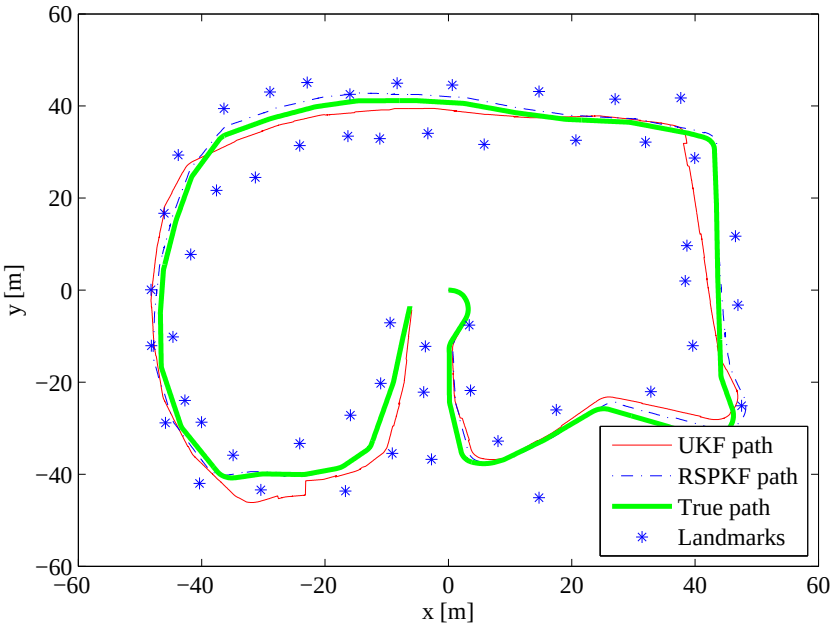


Figure 5.36 : The feature map and the estimated robot path.

In the next part experimental results of the RSPKF and their comparison to UKF is given.

B. Experimental Results in 3D

In this section, Hannover dataset is used and its properties are previously discussed. The proposed RPSKF-SLAM method requires the Gaussian noise; however, the relative Odometry error variation is neither zero-mean nor Gaussian as shown in Figure 5.32. Therefore, the problem becomes more challenging with respect to the

Gaussian case. The vehicle model, observation model, and the augmentation models are presented in the previous section, so they are not given in this section.

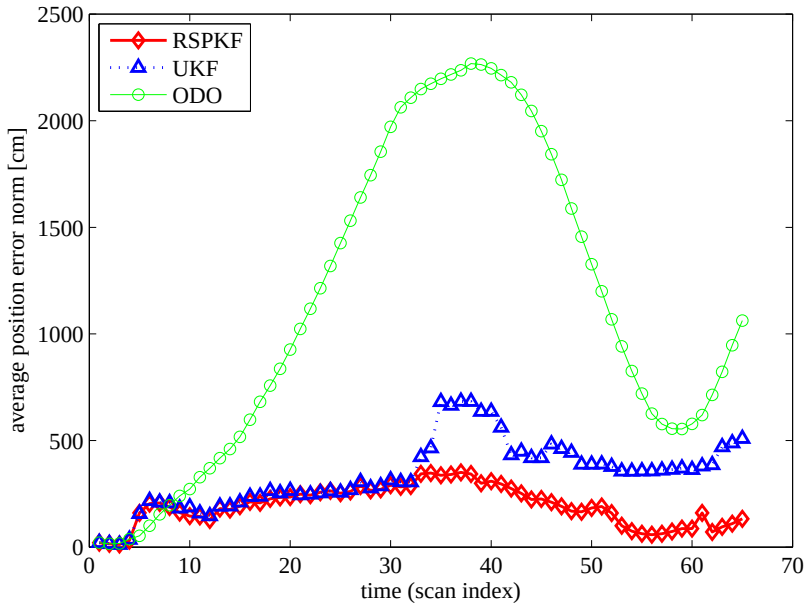


Figure 5.37 : Average position error norm.

The error norm of the robot position and orientation in three dimensions are shown in Figure 5.37 and Figure 5.38, where the process noise covariance matrix $\mathbf{Q} = diag(\sigma_x^2, \varepsilon^2, \sigma_z^2, \varepsilon^2, \sigma_\theta^2, \varepsilon^2)$ is set by $\sigma_x = \sigma_y = 25\,\text{cm}$ and $\sigma_\theta = 0.1°$. The measurement noise covariance matrix $\mathbf{R} = diag(\sigma_{n_x}^2, \sigma_{n_y}^2, \sigma_{n_z}^2, \sigma_d^2)$ is taken as constant and is set by $\sigma_n^2 = 0.001$ and $\sigma_d^2 = 10$ cm.

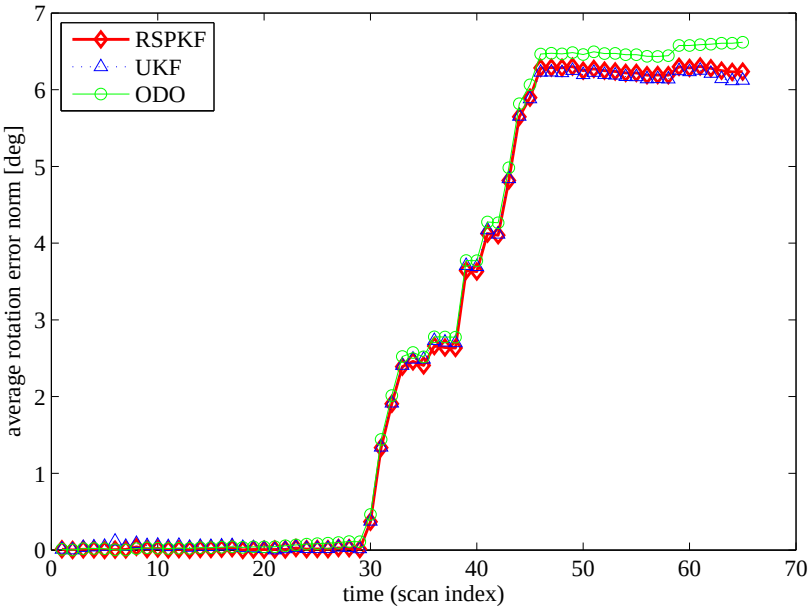


Figure 5.38 : Average rotation error norm

The results show that UKF and CKF based SLAM satisfy the similar results with a maximum position error around 6.5 meters (in the 35 time index). On the other hand, the RSPKF based SLAM has a maximum of 4 meter position error norm and more accurate than the conventional sigma point approaches. The rotation error norm looks similar for all filter types.

In the next section, planar-feature based SLAM with Sparse Extended Information Filters (SEIF) is given.

5.2.1.4 Performance Evaluations of the SEIF Based SLAM

Due to the dual representations of the covariance and information forms, while the time update is complex for information filters, the measurement update is complex for Kalman filters. For this reason, the information matrix is sparsified by deactivating the passive features links, and sparse information filters are obtained. Therefore, a scalable method for large scale SLAM is obtained since a constant time solution is obtained for the motion update step of the information filters. In this section, first, the measurement and update times of SEIF filters are investigated. Then the position error norms of the SEIF based SLAM are given.

The measurement and motion updates process times of the SEIF based SLAM are given in Figure 5.39 and Figure 5.40, respectively.

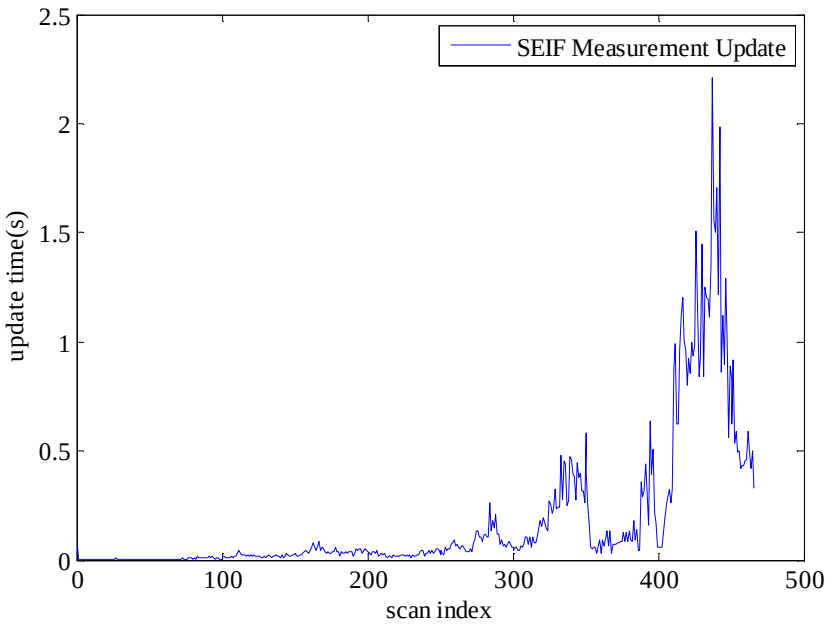


Figure 5.39 : SEIF measurement updates processing time.

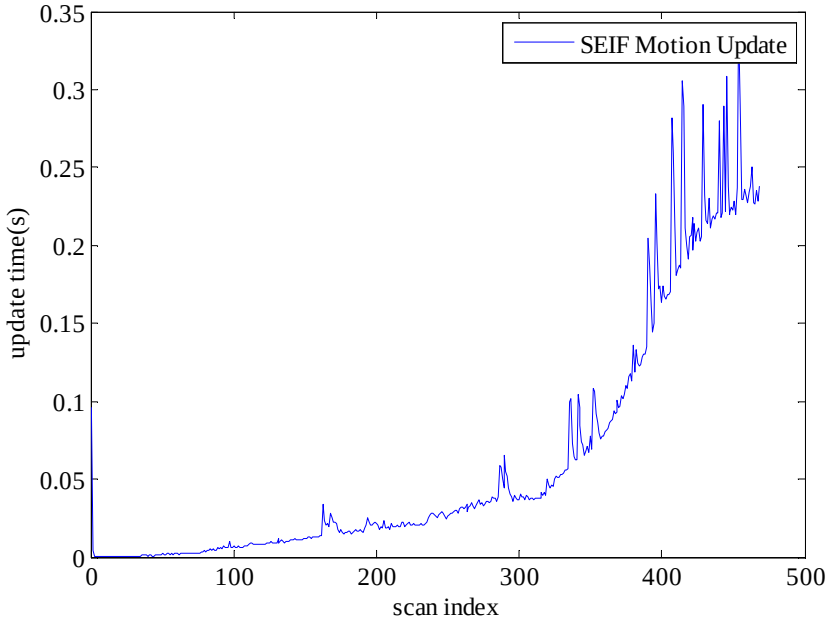


Figure 5.40 : SEIF motion updates processing time.

For comparisons, the update times of the EKF filters are also given in Figure 5.41 and Figure 5.42. If the measurment and motion updates processing times of EKF and SEIF are compared, the significant improvement is obtained. In conventional point feature based SLAM, the measurment update times of EKF filter is quadratically proportional to the size of the state vector. However, In plane-feature based SLAM, it is observed the update times are decreased significantly during the loop closure. The reason of this is that the planes are merged when they associated after the measurment updated.

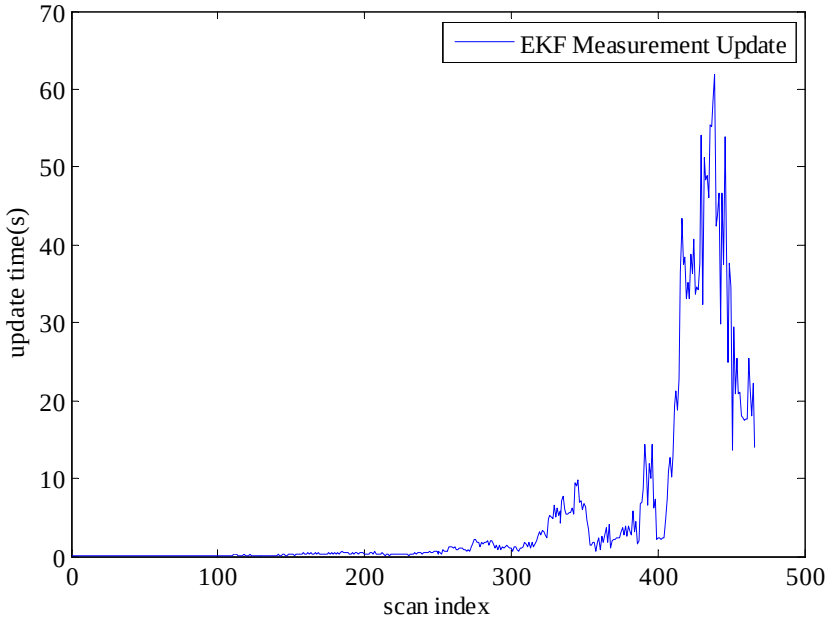


Figure 5.41 : EKF measurement updates processing time.

It can also be obvious said that the motion update processing times of SEIF and EKF has some difference, but when compared to measurment update processing times, they can be ignored. The position error variation of SEIF based SLAM is given in Figure 5.43. At the end of the movement, which corresponds to 468th scan index, the error on the position of the SEIF based SLAM is about 90 cm, which is very satisfactory when compared to odometry based error.

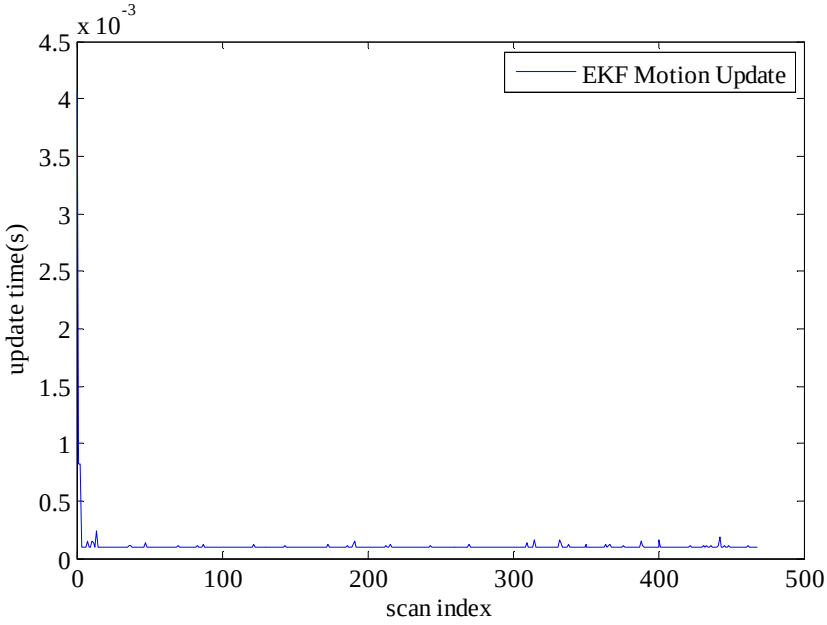


Figure 5.42 : EKF motion updates processing time.

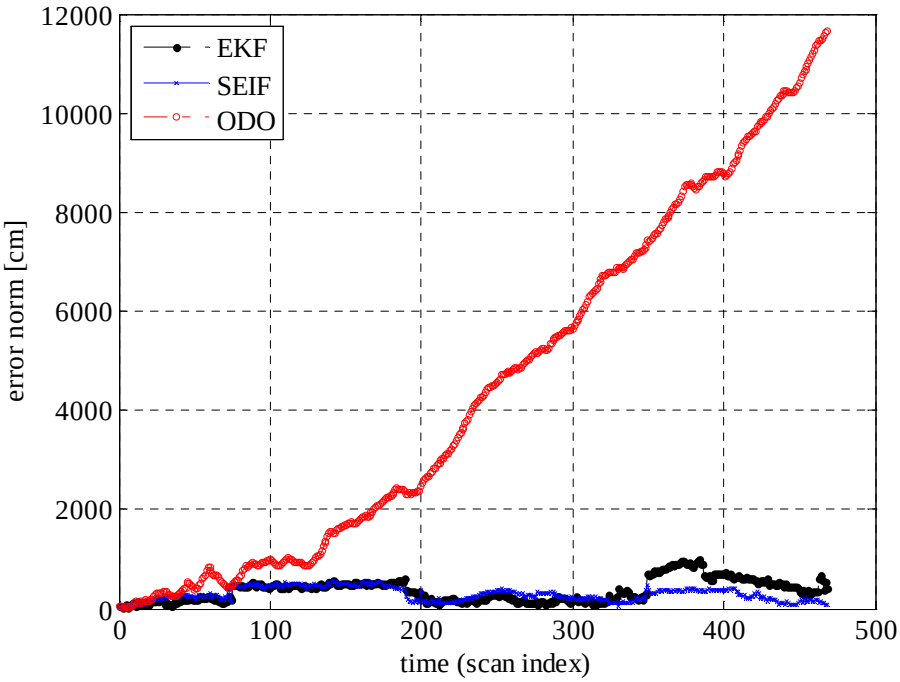


Figure 5.43 : Position error norm of SEIF based SLAM.

In Figure 5.44, the estimated robot trajectory with SEIF based SLAM and EKF based SLAM are shown and compared to ground truth. The results show that SEIF based SLAM outperforms the EKF as seen in the left-bottom of the map. A second illustration on the SEIF is that the image of information matrix with and without sparsification.

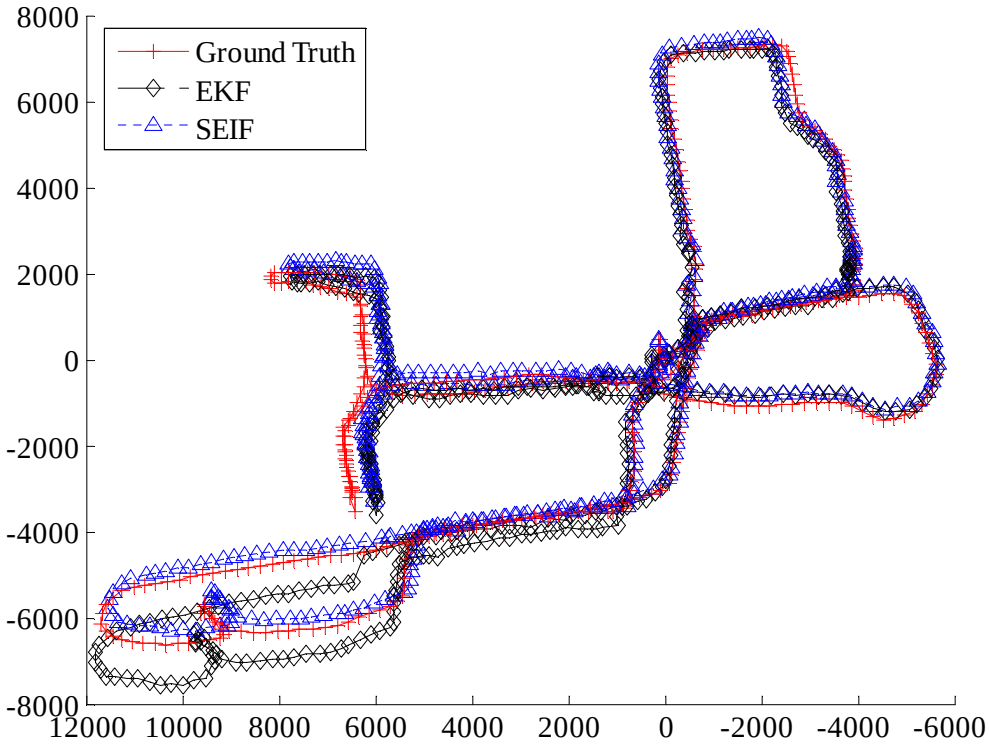
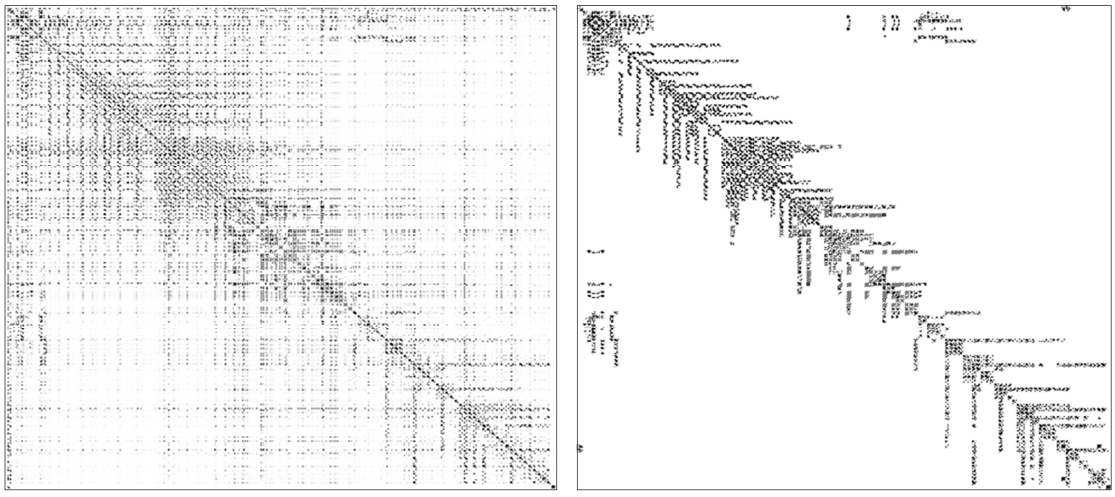


Figure 5.44 : Estimated robot trajectory in xz projection.



(a) (b)

Figure 5.45 : An illustration of a small information matrix: (a)without sparsification. (b)with sparsification.

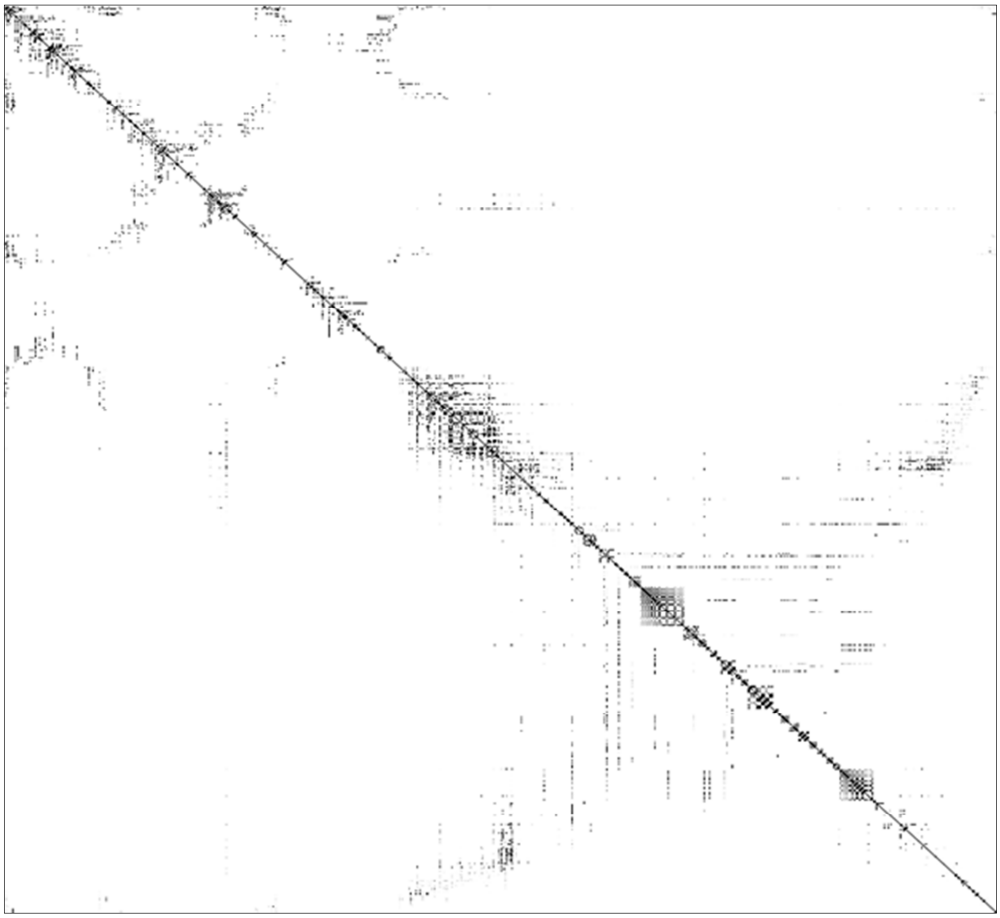


Figure 5.46 : Sparse information matrix including 743 features.

In Figure 5.45(a) shows an image of a small information matrix without sparsification, which includes the robot pose and 118 features. Here, the information matrix size is 478×478. Figure 5.45(b) shows the image of a small information matrix with sparsification, which also holds the 118 features as in the previous case. As seen, the week links are eliminated, and a sparse matrix is obtained.

For the whole map, holding 743 planar-features, in the Hannover dataset, the sparse information matrix with the features and robot pose is shown Figure 5.46. The size of the information matrix in this case is 2976×2976.

5.3 Discussions

In this section, the experiment results are given for both the scan matching and feature based SLAM methods. In the first part, the performance of a new scan matching method based on normal distribution transform, ML-NDT, is elucidated.

Then the performance of FbML-NDT, which combines the planar-feature extraction method with scan matching in a framework, is presented. In the second part, the planar-feature based SLAM methods are investigated. The performance of the Gaussian filters, which are EKF, UKF, CKF, RSPKF, SEIF are analyzed. The results show that planar-feature extraction method improves the performance of both scan matching and 6D-SLAM considerably.

6. BENCHMARKING METHODS IN 6D SLAM

In this chapter, two SLAM benchmarking methods are discussed, and the results are given for the introduced filtering method in Chapter 5. The first benchmarking method is proposed by Wulf et al. [74], and the second method is proposed by Kümmerle et al. [27]. Firstly, the methods are elucidated, and then the benchmarking results are given for the feature based SLAM.

6.1 Benchmarking Method 1

Large number of studies is proposed to solve SLAM problem by now. However, since the ground truth data is not available, it is significant to compare different SLAM approaches objectively. For that purpose, Wulf et al. present a benchmarking method for generating the ground truth data based on reference maps [74]. This method is based on the final SLAM results and a reference position obtained independently of the SLAM algorithm under test. Since the RTK-GPS receivers cannot be used in urban outdoor environments, they proposed a technique that is based on surveyed maps, which can be obtained from a registry office of the country land, where the country is German in the seminal paper. The process of obtaining the reference positions of ground truth data has two steps. Firstly, Monte Carlo Localization (MCL) step matching the sensor data to the highly accurate map is applied. Secondly, a manual quality control is carried out to validate the MCL results.

The benchmarking criteria is based on the objective performance given by (6.1) for position and orientation independently.

$$\sigma_{Wulf} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} e_i^2} \qquad \textbf{(6.1)}$$

where e_i can be either position or orientation as in (3.1) and (6.3) .

$$e_{pos,i} = \sqrt{(x_i^{SLAM} - x_i^{REF})^2 + (y_i^{SLAM} - y_i^{REF})^2 + (z_i^{SLAM} - z_i^{REF})^2} \quad \textbf{(6.2)}$$

$$e_{ori,i} = \sqrt{(\alpha_i^{SLAM} - \alpha_i^{REF})^2 + (\beta_i^{SLAM} - \beta_i^{REF})^2 + (\gamma_i^{SLAM} - \gamma_i^{REF})^2} \quad \textbf{(6.3)}$$

In the next section, the second benchmarking method proposed by Kümmerle et al. is explained.

6.2 Benchmarking Method 2

In this section, a benchmarking metric introduced by Kümmerle is used for measuring the performance of SLAM approaches not by comparing the map itself but by considering the poses of the robot during the data acquisition [27]. Thus, the method has two important advantages: First, it allows comparing the results of algorithms generating different types of map metric map representations. Second, the method is invariant to sensor setup of the vehicle. In this way, a result of a graph-based SLAM approach working on laser range data can be compared with the result of vision-based FastSLAM. The only required property is that the SLAM algorithm estimates the trajectory of the robot given by a set of poses where the observations are made. All benchmark computations are performed on this set of poses.

Let $x_{1:T}$ be the poses of the robot estimated by a SLAM algorithm from time step 1 to *T*. The reference poses of the robot is denoted by $x_{1:T}^*$. An error metric can be defined as

$$\varepsilon(x_{1:N}) = \sum_{i=1}^{N} x_i \ominus x_i^* \quad \textbf{(6.4)}$$

where $\oplus$ is the motion composition operator and $\ominus$ its inverse. The relative transformation is defined as $\delta_{i,j} = x_j \ominus x_i$ which moves the robot from position x_i to x_j. Based on this definition, (6.4) can be rewritten as

$$\varepsilon(x_{1:N}) = \sum_{i=1}^{N} \left((x_i \oplus \delta_{1,2} \oplus \ldots \oplus \delta_{N-1,N}) \ominus (x_1^* \oplus \delta_{1,2}^* \oplus \ldots \oplus \delta_{N-1,N}^*) \right)^2 \quad \textbf{(6.5)}$$

In the paper, it is claimed that this metric is suboptimal for comparing the result of a SLAM algorithm. For example, consider a 1D example where a robot moves along a straight line and let the robot makes a translational error *e* during the first motion,

$\delta_{1,2} = \delta^*_{1,2} + e$, and exact estimates at all the remaining motions, $\delta_{i,i+1} = \delta^*_{i,i+1}$. Hence, the error $\varepsilon(x_{1:N})$, will be *T.e*, since $\delta_{1,2}$ is contained in every pose estimate. However, if the robot makes an error at the end of the motion, $\delta_{N-1,N} = \delta^*_{N-1,N} + e$ and the all others are exact estimates, then it is obtained an error of *e* only. For this reason, this metric is suboptimal for comparing the result of SLAM algorithms. Instead, a novel metric based on the relative displacement between robot poses is proposed as

$$\sigma_{Kümmerle}(\delta) = \sqrt{\frac{1}{N}\sum_{i,j} trans(\delta_{i,j} \ominus \delta^*_{i,j})^2 + rot(\delta_{i,j} \ominus \delta^*_{i,j})^2} \qquad \textbf{(6.6)}$$

where *trans*(·) and *rot*(·) are used to separate and weight the translation and rotational components, and practically they should be evaluated individually.

Again, the true relative displacements between poses are obtained by using the information recorded by the mobile robot and the background knowledge of the human recording the dataset, which requires the manual work.

6.3 Evaluation of Benchmarking Methods For Online SLAM

In this section, the benchmarking methods proposed in the previous sections are evaluated for online planar feature based SLAM. To be able to compare two methods, they are investigated by using two simple examples in 1D and 2D.

In 1D example, assume that the robot has three different cases as shown in Figure 6.1. In this example, robot travels along the *x*-axis with 4 units intervals with three movements, and the true position is denoted by x^*_i. In Case 1, the robot only makes 3 units error at the beginning of the motion and the other estimates are exactly true. In Case 2, the robot only makes 1 unit error during the second motion and other estimates are perfect. Finally, the robot makes 1 unit error at the end of motion, and the others are true. For the three cases, the two metrics are computed and the results are given in Table 6.1.

The metric σ_{Wulf} is computed as

$True: x_i^*$	
$Case1: x_i^1$	
$Case\,2: x_i^2$	
$Case\,3: x_i^3$	

Figure 6.1 : 1D Example for benchmarking comparisons

$$\sigma_{Wulf}^1 = \sqrt{\frac{1}{4}(0^2+3^2+0^2+0^2)} = 1.5 \quad \textbf{(6.7)}$$

$$\sigma_{Wulf}^2 = \sqrt{\frac{1}{4}(0^2+1^2+0^2+0^2)} = 0.5 \quad \textbf{(6.8)}$$

$$\sigma_{Wulf}^3 = \sqrt{\frac{1}{4}(0^2+0^2+0^2+1^2)} = 0.5 \quad \textbf{(6.9)}$$

The metric $\sigma_{Kümmerle}$ is based on relative transformations and they are computed as

$$\delta_{1,2}^* = \delta_{2,3}^* = \delta_{3,4}^* = 4 \quad \textbf{(6.10)}$$

$$\delta_{1,2}^1 = 4,\ \delta_{2,3}^1 = 7,\ \delta_{3,4}^1 = 4 \quad \textbf{(6.11)}$$

$$\delta_{1,2}^2 = 5,\ \delta_{2,3}^2 = 3,\ \delta_{3,4}^2 = 4 \quad \textbf{(6.12)}$$

$$\delta_{1,2}^3 = 4,\ \delta_{2,3}^3 = 4,\ \delta_{3,4}^3 = 5 \quad \textbf{(6.13)}$$

The metric σ_{Wulf} is found as

$$\sigma_{Kümmerle}^1 = \sqrt{\frac{1}{3}\left((4-1)^2+(7-4)^2+0^2\right)} = 2.45 \quad \textbf{(6.14)}$$

$$\sigma_{Kümmerle}^2 = \sqrt{\frac{1}{3}\left((5-4)^2+(3-4)^2+0^2\right)} = 0.82 \quad \textbf{(6.15)}$$

$$\sigma_{Kümmerle}^3 = \sqrt{\frac{1}{3}\left((4-4)^2+(4-4)^2+(5-4)^2\right)} = 0.58 \quad \textbf{(6.16)}$$

As seen from the Table 6.1, although the robot makes the same error in the second and third motion, which is one-third of the error of the first motion, the Kümmerle metric $\sigma_{Kümmerle}$ produces inconsistent result.

Table 6.1 : Comparison of benchmarking methods in 1D.

	σ_{Wulf}	$\sigma_{Kümmerle}$
$Case1: x_i^1$	1.5	2.45
$Case2: x_i^2$	0.5	0.82
$Case3: x_i^3$	0.5	0.58

The second example is in 2D, and two cases are given in Figure 6.2. In Case 1, the robot makes a large error at the first movement and corrects it in the second one to robot true position. In Case 2, robot makes half error in the estimation of second motion with respect to Case 1.

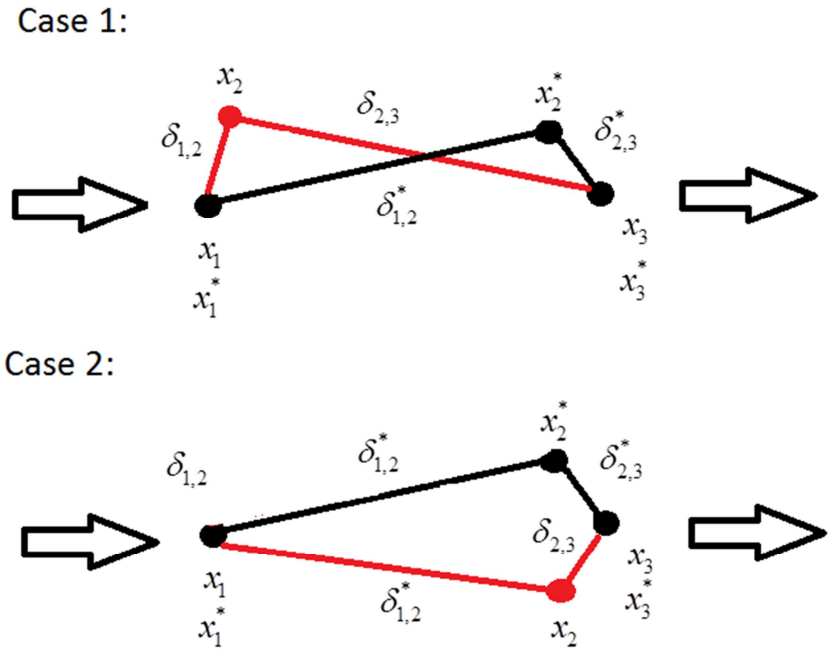


Figure 6.2 : 2D example for benchmarking comparisons

The computations for the metric σ_{Wulf} are given as

$$x_1 = x_1^* = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad x_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad x_2^* = \begin{bmatrix} 5 \\ 1 \end{bmatrix}, \quad x_3 = x_3^* = \begin{bmatrix} 6 \\ 0 \end{bmatrix} \tag{6.17}$$

$$x_2^* - x_2 = \begin{bmatrix} 5 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \end{bmatrix} \Rightarrow d = \sqrt{4^2 + 0^2} = 4 \tag{6.18}$$

$$\begin{aligned} \sigma_{Wulf}^1 &= \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i^* - x_i)^2} = \sqrt{\frac{1}{3}\left((x_1^* - x_1)^2 + (x_2^* - x_2)^2 + (x_3^* - x_3)^2\right)} \\ &= \sqrt{\frac{1}{3}}\sqrt{(x_2^* - x_2)^2} = \sqrt{\frac{d^2}{3}} = \sqrt{\frac{4^2}{3}} = 2.31 \end{aligned} \tag{6.19}$$

The computations for the $\sigma_{Kümmerle}$ is given as

$$\delta_{1,2}^* = \begin{bmatrix} 5 \\ 1 \end{bmatrix},\ \delta_{1,2} = \begin{bmatrix} 1 \\ 1 \end{bmatrix},\ \delta_{2,3}^* = \begin{bmatrix} 1 \\ -1 \end{bmatrix},\ \delta_{2,3} = \begin{bmatrix} 5 \\ -1 \end{bmatrix} \tag{6.20}$$

$$\begin{aligned} \delta_{1,2}^* - \delta_{1,2} &= \begin{bmatrix} 5 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \end{bmatrix} \Rightarrow d_{1,2} = 4 \\ \delta_{2,3}^* - \delta_{2,3} &= \begin{bmatrix} 1 \\ -1 \end{bmatrix} - \begin{bmatrix} 5 \\ -1 \end{bmatrix} = \begin{bmatrix} -4 \\ 0 \end{bmatrix} \Rightarrow d_{2,3} = 4 \end{aligned} \tag{6.21}$$

$$\begin{aligned} \sigma_{Kümmerle}^1 &= \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\delta_{i,j}^* - \delta_{i,j})^2} = \sqrt{\frac{1}{2}((\delta_{1,2}^* - \delta_{1,2})^2 + (\delta_{2,3}^* - \delta_{2,3})^2)} \\ &= \sqrt{\frac{(\delta_{1,2}^* - \delta_{1,2})^2 + (\delta_{2,3}^* - \delta_{2,3})^2}{2}} = \sqrt{\frac{d_{1,2}^2 + d_{2,3}^2}{2}} = 4 \end{aligned} \tag{6.22}$$

The computations of σ_{Wulf}^1 for the Case 2,

$$x_1 = x_1^* = \begin{bmatrix} 0 \\ 0 \end{bmatrix},\quad x_2 = \begin{bmatrix} 5 \\ -1 \end{bmatrix},\quad x_2^* = \begin{bmatrix} 5 \\ 1 \end{bmatrix},\quad x_3 = x_3^* = \begin{bmatrix} 6 \\ 0 \end{bmatrix} \tag{6.23}$$

$$x_2^* - x_2 = \begin{bmatrix} 5 \\ 1 \end{bmatrix} - \begin{bmatrix} 5 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \end{bmatrix} \Rightarrow d = \sqrt{2^2 + 0^2} = 2 \tag{6.24}$$

$$\begin{aligned} \sigma_{Wulf}^2 &= \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i^* - x_i)^2} = \sqrt{\frac{1}{3}\left((x_1^* - x_1)^2 + (x_2^* - x_2)^2 + (x_3^* - x_3)^2\right)} \\ &= \sqrt{\frac{1}{3}(x_2^* - x_2)^2} = \sqrt{\frac{d^2}{3}} = \sqrt{\frac{2^2}{3}} = 1.15 \end{aligned} \tag{6.25}$$

The computations of $\sigma_{Kümmerle}^2$ for the Case 2,

$$\delta_{1,2}^* = \begin{bmatrix} 5 \\ 1 \end{bmatrix},\ \delta_{1,2} = \begin{bmatrix} 5 \\ -1 \end{bmatrix},\ \delta_{2,3}^* = \begin{bmatrix} 1 \\ -1 \end{bmatrix},\ \delta_{2,3} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{6.26}$$

$$\delta^*_{1,2} - \delta_{1,2} = \begin{bmatrix} 5 \\ 1 \end{bmatrix} - \begin{bmatrix} 5 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \end{bmatrix} \Rightarrow d_{1,2} = 2$$
$$\delta^*_{2,3} - \delta_{2,3} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -2 \end{bmatrix} \Rightarrow d_{2,3} = 2 \quad \textbf{(6.27)}$$

$$\sigma^2_{Kümmerle} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\delta^*_{i,j} - \delta_{i,j})^2} = \sqrt{\frac{1}{2}((\delta^*_{1,2} - \delta_{1,2})^2 + (\delta^*_{2,3} - \delta_{2,3})^2)}$$
$$= \sqrt{\frac{(\delta^*_{1,2} - \delta_{1,2})^2 + (\delta^*_{2,3} - \delta_{2,3})^2}{2}} = \sqrt{\frac{d^2_{1,2} + d^2_{2,3}}{2}} = 2 \quad \textbf{(6.28)}$$

The comparisons of both benchmarking method is seen in Table 6.2. In two examples, it is seen that the Kümmerle metric provides large values with respect to Wulf metric. A possible reason of this phenomenon is that when a robot makes a large relative error when estimating the robot position at the previous motions, then it has to make large relative transformations to correct the robot position since the main goal is to track the robot position. Thus, the Kümmerle metric mostly generate large values when this kind of situations occurs. As consequence of this phenomenon, the individual error variations producing the metric mostly have large values while closing the loops.

Table 6.2 : Comparison of benchmarking methods in 2D.

	σ_{Wulf}	$\sigma_{Kümmerle}$
$Case1: x_i^1$	2.31	4
$Case2: x_i^2$	1.15	2

6.4 Benchmarking Results

In this section, the results of planar-feature based SLAM algorithms based on two metrics, Kümmerle and Wulf, are given. The Hannover dataset, containing 468 3D scans is used for benchmarking. The dataset [73] are also used for performance comparisons in the previous section, and the same dataset is used in benchmarking for consistency.

The next two sections present the benchmarking results of planar- feature based SLAM methods.

6.4.1 Benchmarking Results with Wulf Metric

In this section, the feature based SLAM methods are evaluated and their benchmarking results are given. EKF based SLAM error variations for 468 scans are shown in Figure 6.3 for translations and rotations, and the Wulf metric for both rotation and translation is computed. The Wulf metric is computed for translation σ_T and rotation σ_R independently and shown on the same figure.

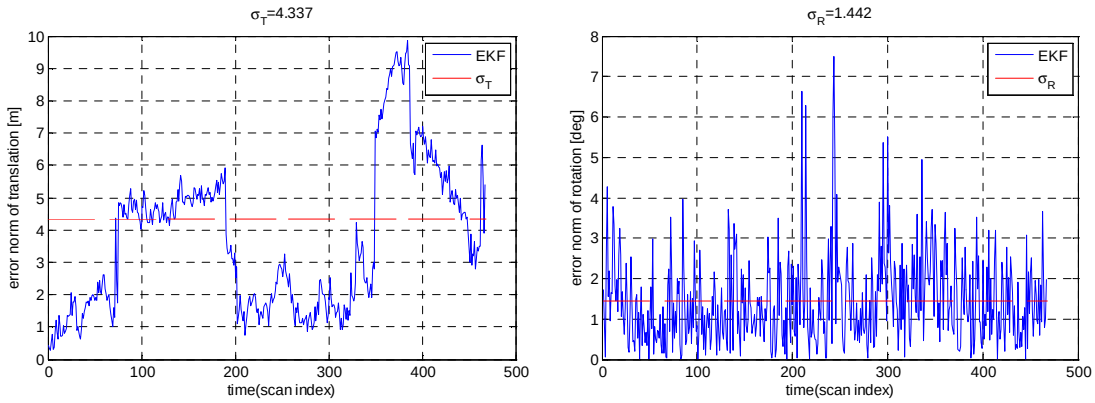


Figure 6.3 : EKF translation and rotation error norms and Wulf metric.

Similarly, the UKF based SLAM error variations for the same number of scans are shown in Figure 6.4 for translations and rotations, and the Wulf metric for both rotation and translation is calculated.

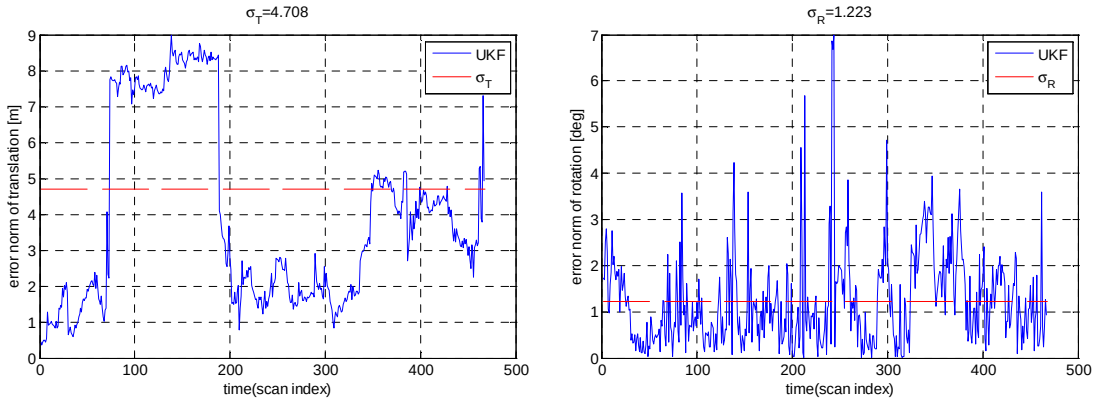


Figure 6.4 : UKF translation and rotation error norms and Wulf metric.

CKF based SLAM error variations for the same number of scans are shown in Figure 6.5 for translations and rotations, and the Wulf metric for both rotation and translation is found.

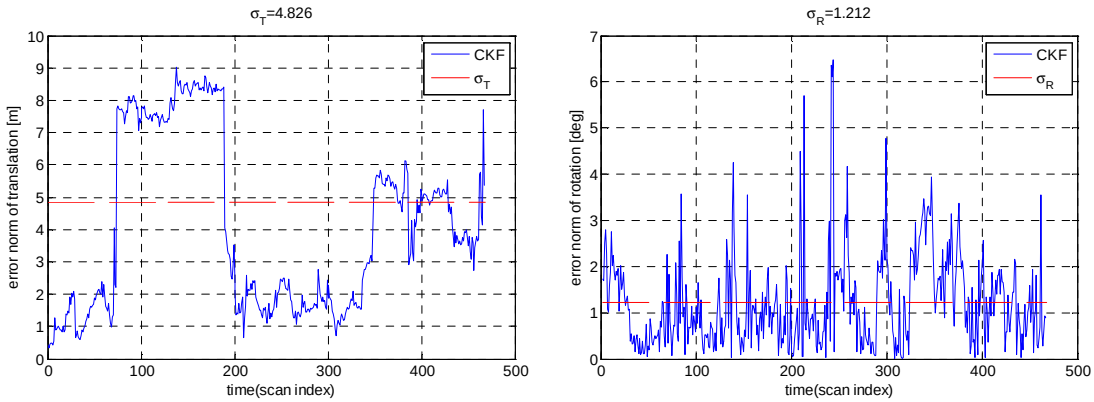


Figure 6.5 : CKF translation and rotation error norms and Wulf metric.

Finally, SEIF based SLAM error variations for the same number of scans are shown in Figure 6.6 for translations and rotations, and the Wulf metric for both rotation and translation is calculated.

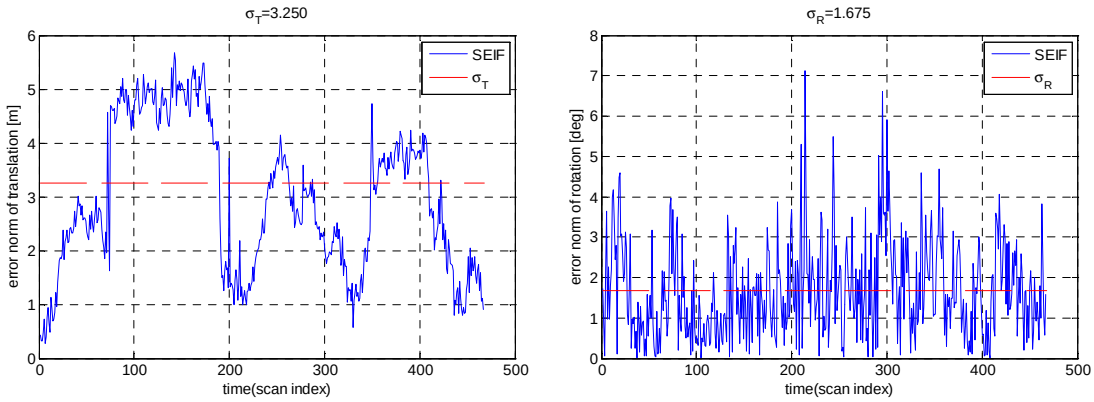


Figure 6.6 : SEIF translation and rotation error norms and Wulf metric.

The results show that the best performance is obtained by EKF for rotations and SEIF for translation. However, since the error in rotation estimation is already small, the SEIF performance is very satisfactory when its other advantages are taken into consideration.

In the next section the benchmarking results of the planar-feature based SLAM with Kümmerle metric is discussed.

6.4.2 Benchmarking Results with Kümmerle Metric

First, EKF based SLAM relative error variations for 468 scans are shown in Figure 6.7 for translations and rotations, and the Kümmerle metric for both rotation and translation is computed. The Kümmerle metric is computed for translation σ_T and rotation σ_R independently and shown on the same figure.

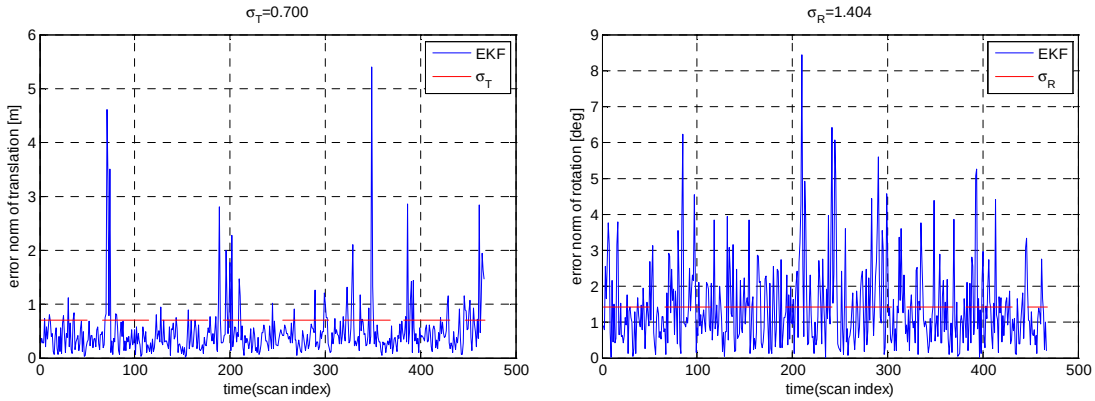


Figure 6.7 : EKF translation and rotation error norms and Küm. metric.

UKF based SLAM relative error variations for the same number of scans are shown in Figure 6.8 for translations and rotations, and the Kümmerle metric for both rotation and translation is found.

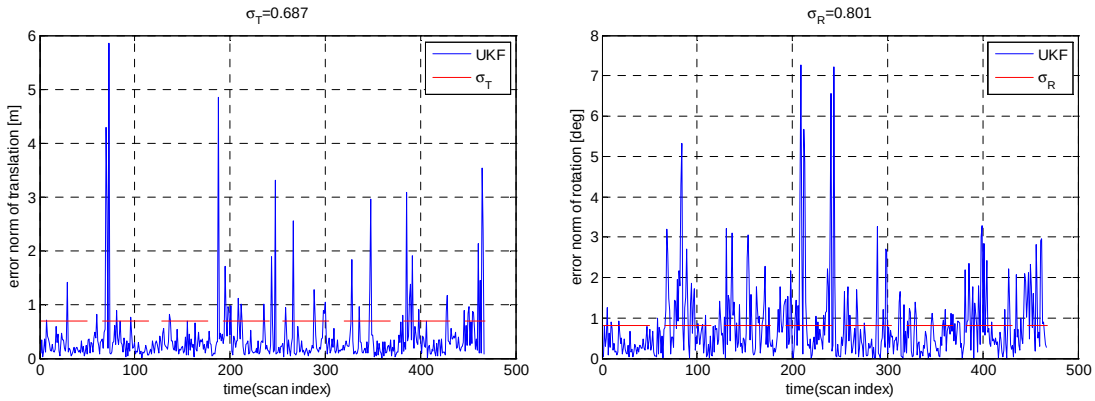


Figure 6.8 : UKF translation and rotation error norms and Küm. metric.

Likewise, the CKF based SLAM relative error variations for the same number of scans are shown in Figure 6.9 for translations and rotations, and the Kümmerle metric for both rotation and translation is obtained.

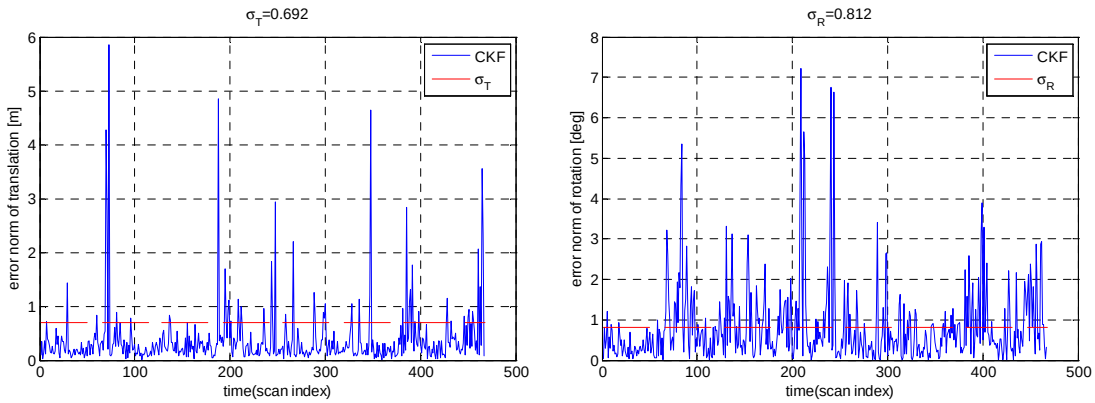


Figure 6.9 : CKF translation and rotation error norms and Küm. metric.

Finally, SEIF based SLAM relative error variations for the same number of scans are shown in Figure 6.10 for translations and rotations, and the Kümmerle metric for both rotation and translation is computed.

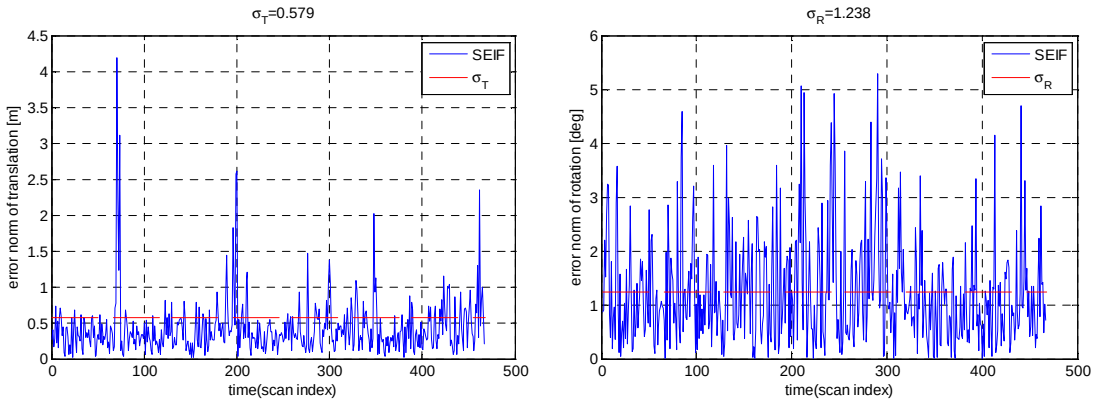


Figure 6.10 : SEIF translation and rotation error norms and Küm. metric.

The results show that the best performance is obtained by SEIF for translations and EKF for rotations. An important property of the Kümmerle metric should be noted that the relative translation error variations are significantly increases while the loop is being closed, where they can be seen at the 70^{th}, 200^{th}, 350^{th}, and the 460^{th} scans.

6.5 Discussions

In this section, two existing benchmarking methods proposed by Kümmerle and Wulf is introduced and based on these methods, the proposed feature based SLAM performances by this study are compared. From the experiments, it can be inferred that the benchmarking method of Wulf is more suitable for measuring the performance of SLAM methods since Kümmerle methods generates high errors when loop closing. For online feature based SLAM, this high relative errors are normal while loop closing since the mobile robot force itself to catch the true position by re-observing and associating the features in the map. On the other hand, the Wulf's method specifically measures the error on the position estimate, and thus; his metric provides better benchmarking performance than Kümmerle's metric for online feature based SLAM.

7. CONCLUDING REMARKS

This dissertation is concluded with a summary of the work and suggestions for the future research.

7.1 Summary

The research presented in this thesis provides various solutions to simultaneous localization and mapping problem. First, a scan registration algorithm based on Normal Distribution Transform is introduced. This algorithm uses multi-layered structure that automatically calculates the cell size from the point cloud boundaries instead of assigning a fixed cell size. In each layer, the point cloud is divided into q^n cells, where n is the layer number and q is the division constant. Then for each cell, the conventional NDT generative process is applied, the mean and covariance values of each cell containing more than four points are calculated and stored. Moreover, a different score function than the original method is used, and Levenberg-Marquardt optimization method is adapted to the algorithm in addition to Newton method. An extended comparison of the NDT and ML-NDT, especially in 3D outdoor, is given for the experimental datasets. The performance of the optimization methods are also compared with each other. In addition, the effects of the sampling strategies are analyzed, and an easy to use sampling method, without extra computation, is proposed. The results show that the proposed method provides fast convergence rate and long-range measurement capabilities than NDT. Finally, the application of ML-NDT to the SLAM problem in terms of 2D simulation data and 3D experimental data is given.

Second, a 3D robust feature extraction method for outdoor and large scale SLAM is proposed. In this method, after discretizing the point cloud into regular cell, plane segments are found. Then horizontal planes are eliminated to discard ground effect. At the end, the filtered planes are merged using a plane-merging algorithm. In merging step, an additional condition is added to satisfy the closeness of two planes with respect to conventional methods. In addition, we proposed a PCA based

projection method. The extracted plane properties give the opportunity to solve data association problem easily. Since the obtained features are planar, the conventional data association methods for point features cannot be used. There, the semantic data obtained from the feature extraction method is also used in the correspondence decision.

Third, a fast and robust scan matching method based on the proposed feature extraction is proposed, and the effect of sampling strategies on feature based scan matching is investigated. The scan matching part is based on multi-layered Normal Distribution Transform, and the feature extraction part is based on a plane detection method. These two methods are combined in such a way that the extracted plane inlier points are used in the registration process of the scan matching algorithm. In this way, it is obtained a robust and fast both scan matching and feature extraction algorithms. From the scan matching point of view, the this method is robust because the matching is based on the certain geometric structures like planar surfaces and faster with respect to the conventional methods since the number of matching points is reduced effectively and efficiently, which is very important due to the iterative structure of the scan matching method.

Forth, the extracted planar-features are proposed for feature based SLAM and its application to 3D outdoor SLAM is presented. The method is adapted to filters EKF, UKF, CKF, RSPKF, and SEIF to solve feature based SLAM problem for large-scale and outdoor environments. The performance results are compared for the experimentally obtained dataset. The results show that all these filters can be used with the planar features successfully.

Finally, a data association method for the planar-features is proposed. The method is based on the rich plane properties, considered as semantic data, obtained by the feature extraction method. The data association method uses the same criteria used in the plane merging criteria for the decision of correspondence of features. This is an important contribution since the conventional data association methods are for point features and its computational complexity increases exponentially with the number of feature, while the semantic data association proposed in this thesis has linear complexity. In addition, the traditional data association methods cannot be used for planar features since planes are encoded in the state vector in infinite form, so they

cannot be distinguished with the simple mean and covariance values of the parameters of plane normal and minimum distance to origin.

7.2 Future Directions

The main purpose of this dissertation was to propose a six-dimensional simultaneous localization and map building method for large-scale and outdoor environments; therefore, the development of methodology was the central objective of this thesis. The implementation of the proposed methods in a real-time system, which is under construction in the Robotic Laboratory of Control Engineering Department of the Istanbul Technical University, is the primary feature direction. Next, there might be a few improvements on the proposed methods seen by the author, and they are discussed below.

Lu and Milios [31] presented a globally consistent solution to the simultaneous localization and mapping problem in 2D with three degrees of freedom poses, and its extension to 3D SLAM problem using Iterative Closest Point (ICP) algorithm with six degrees of freedom is proposed in [6]. Therefore, the performance of the proposed scan matching methods, ML-NDT and FbML-NDT, should be tested with this globally consistent 3D mapping algorithm.

In general, it is seen that the SLAM methods are separated into two parts in terms of their mapping model, such as feature or volumetric data, which can be laser scan or camera image depending on the sensor technology used. The difficult part of the feature based SLAM is to obtain robust and limited number of features for large-scale and outdoor SLAM. In this thesis, a planar-feature extraction method was proposed for this purpose. On the other hand, the difficult part of the scan matching based SLAM methods is the “loop closure” or “loop detection” problem, and the detection of the loop cannot be achieved with the individual scans since they contain very dense data. For that reason, the SLAM methods utilizing scan matching, also require an independent loop detection mechanism, which is mostly based on other type of measurement system like camera and a feature detection method, to be able to obtain globally consistent solution. As another feature study, the proposed scan matching methods can be integrated to planar-feature based SLAM method. In this framework, the scan matching method, which provides more accurate result than the odometry and inertial measurement unit sensors, should be used to find the

prediction of robot pose in the motion update step of the feature based SLAM method. This is also useful for semantic data association proposed by this thesis since it depends on the robot current location, and it can be obtained by using rigid body transformation parameters from the scan matching. Most importantly, the feature extraction and feature based scan matching method (FbML-NDT), which is robust and fast in nature, have very common parts in terms of computations. In this regard, the combination of both methods does not introduce a significant computational complexity. Therefore, a more accurate and high-speed hybrid SLAM algorithm can be obtained, which has been never applied before, according to the authors knowledge.

In Chapter 4, the planar feature extraction method is proposed, and it is application to feature based SLAM methods are given for several filters, which are all known as Gaussian filters. Since the proposed feature extraction method will be working on outdoor environments, one can never guarantee the type of the noise but assume; therefore, this study can be extended for non-Gaussian filters. A well-known non-Gaussian filters in the solution of feature based SLAM is Particle Filters (PFs) proposed by Montemerlo [37]. However, when the problems of particle filters in large-scale SLAM is considered, a combined method [69] using Sparse Extended Information Filters (SEIF) and PFs can be applied.

In the experimental results, it was shown that the SEIF provides faster computation than the other types of filters. However, it has been shown that SEIF sparsification procedure yields overconfident error estimates when expressed in the global reference frame. Matthew proposed Exactly Sparse Extended Information Filters (ESEIF) [67] as an improved scalable method, which maintains both sparsity and consistency in large-scale SLAM problems. Therefore, this thesis can be extended as the basis of the development of a framework for the ESEIF based SLAM.

REFERENCES

[1] **Arasaratnam, I. and Haykin, S.** (2009). Cubature Kalman Filters. *IEEE Transactions on Automatic Control*, 54(6), 1254-1269.

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

[3] **Biber, P. and Strasser, W.**, The normal distributions transform: a new approach to laser scan matching. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, 2003, pp. 2743-2748 vol.3.

[4] **Biber, P. and Strasser, W.**, nScan-matching: simultaneous matching of multiple scans and application to SLAM. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 2006, pp. 2270-2276.

[5] **Borges, G. A. and Aldon, M. J.** (2004). Line Extraction in 2D Range Images for Mobile Robotics. *J. Intell. Robotics Syst.*, 40(3), 267-297.

[6] **Borrmann, D., Elseberg, J., Lingemann, K., Nuchter, A., and Hertzberg, J.** (2008). Globally consistent 3D mapping with scan matching. *Robot. Auton. Syst.*, 56(2), 130-142.

[7] **Brenneke, C., Wulf, O., and Wagner, B.** (2003). Using 3D laser range data for SLAM in outdoor environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 188-193 vol.1.

[8] **Castellanos, J. A., Montiel, J. M. M., Neira, J., and Tardos, J. D.** (1999). The SPmap: a probabilistic framework for simultaneous localization and map building. *IEEE Transactions on Robotics and Automation*, 15(5), 948-952.

[9] **Castellanos, J. A., Tardos, J. D., and Schmidt, G.** (1997). Building a global map of the environment of a mobile robot: the importance of correlations. In *IEEE International Conference on Robotics and Automation*, pp. 1053-1059 vol.2.

[10] **Deok-Jin, L.** (2008). Nonlinear Estimation and Multiple Sensor Fusion Using Unscented Information Filtering. *IEEE Signal Processing Letters*, 15, 861-864.

[11] **Dissanayake, M., Newman, P., Clark, S., Durrant, H. F., and Csorba, M.** (2001). A solution to the simultaneous localization and map building (SLAM) problem., *IEEE Transactions on Robotics and Automation*, 17(3), 229-241.

[12] **Dunik, J., Straka, O., and Simandl, M.** (2011), The Development of a Randomised Unscented Kalman Filter. Presented at *the Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011.

[13] **Durrant-Whyte, H. F.** (1987). Uncertain geometry in robotics. In *IEEE International Conference on Robotics and Automation*, pp. 851-856.

[14] **Durrant-Whyte, H. F. and Bailey, T.** (2006). Simultaneous localization and mapping: part I. *IEEE Robotics & Automation Magazine*, 13(2), 99-110.

[15] **Durrant-Whyte, H. F., Rye, D., and Nebot, E. M.** (1995). Localisation of automatic guided vehicles. In *Robotics Research: The 7th International Symposium (ISRR'95)*, Springer Verlag, pp. 613-625.

[16] **Fischler, M. A. and Bolles, R. C.** (1981). Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6), 381-395.

[17] **Fraunhofer**. (2010). Available: www.3d-scanner.net

[18] **Genz, A. and Monahan, J.** (1998). Stochastic Integration Rules for Infinite Regions. *SIAM J. Sci. Comput.*, 19(2), 426-439.

[19] **Greenspan, M. and Yurick, M.**, "Approximate k-d tree search for efficient ICP," in *3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on*, 2003, pp. 442-448.

[20] **Guivant, J., Nebot, E., and Baiker, S.** (2000). Localization and map building using laser range sensors in outdoor applications. *Journal of Robotic Systems*, 17(10), 565-583.

[21] **Huang, G. P., Mourikis, A. I., and Roumeliotis, S. I.** (2009). On the complexity and consistency of UKF-based SLAM, in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, 2009, pp. 4401-4408.

[22] **Jinliang, L., Jihua, B., and Yan, Y.** (2010), Study on localization for rescue robots based on NDT scan matching, in *Information and Automation (ICIA), 2010 IEEE International Conference on*, pp. 1908-1912.

[23] **Julier, S., Uhlmann, J., and Durrant-Whyte, H. F.** (2000). A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on Automatic Control*, 45(3), 477-482.

[24] **Julier, S. J. and Uhlmann, J. K.** (1997). New extension of the Kalman filter to nonlinear systems. I1 Kadar, (ed.) *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*. City, pp. 182-193.

[25] **Julier, S. J., Uhlmann, J. K., and Durrant-Whyte, H. F.** (1995), A new approach for filtering nonlinear systems, in *American Control Conference, 1995. Proceedings of the*, 1995, pp. 1628-1632 vol.3.

[26] **Kaminade, T., Takubo, T., Mae, Y., and Arai, T.** (2008), The generation of environmental map based on a NDT grid mapping -Proposal of convergence calculation corresponding to high resolution grid, in

IEEE International Conference on Robotics and Automation, 2008, pp. 1874-1879.

[27] **Kümmerle, R., Steder, B., Dornhege, C., Ruhnke, M., Grisetti, G.,** et al**.** (2009). On measuring the accuracy of SLAM algorithms. *Auton. Robots*, 27(4), 387-407.

[28] **Leonard, J. J. and Durrant-Whyte, H. F.** (1991) Simultaneous map building and localization for an autonomous mobile robot," in Intelligent Robots and Systems. *IEEE/RSJ International Workshop on Intelligence for Mechanical Systems*, pp. 1442-1447 vol.3.

[29] **Leonard, J. J. and Feder, H. J. S.** (2000). A Computationally Efficient Method for Large-Scale Concurrent Mapping and Localization, in Robotics Research, *The Ninth International Symposium (ISRR'99)*. New York: Springer-Verlag, pp. 169–176.

[30] **Leung, C., Shoudong, H., and Dissanayake, G.** (2008). Active SLAM in structured environments, in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 1898-1903.

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

[32] **Magnusson, M.** (2009**).** The Three-Dimensional Normal Distributions Transform - an Efficient Representation for Registration, Surface Analysis, and Loop Detection. *PhD Thesis*, Örebro Univeristy, 2009.

[33] **Magnusson, M., Lilienthal, A., and Duckett, T.** (2007). Scan registration for autonomous mining vehicles using 3D-NDT: Research Articles. *J. Field Robot.*, 24(10), 803-827.

[34] **Magnusson, M., Nuchter, A., Lorken, C., Lilienthal, A. J.,** et al. (2009), Evaluation of 3D registration reliability and speed - A comparison of ICP and NDT, in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, 2009, pp. 3907-3912.

[35] **Martinez-Cantin, R. and Castellanos, J. A.** (2005), Unscented SLAM for large-scale outdoor environments, in *Intelligent Robots and Systems, IEEE/RSJ International Conference on* , pp. 3427-3432.

[36] **McManus, C., Furgale, P., and Barfoot, T. D.** (2011). Towards appearance-based methods for lidar sensors, in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1930-1935.

[37] **Montemerlo, M.** (2003), FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem With Unknown Data Association. *PhD Thesis*, School of Computer Science, Carnegie Mellon University.

[38] **Newman, P., Sibley, G., Smith, M., Cummins, M., Harrison,** et al**.** (2009). "Navigating, Recognizing and Describing Urban Spaces With Vision and Lasers." Int. J. Rob. Res., 28(11-12), 1406-1433.

[39] **Nieto, J., Guivant, J., and Nebot, E.** (2002), FastSLAM: Real Time Implementation in Outdoor Environments. in *Australasian Conference on Robotics and Automation*, Auckland, 2002.

[40] **Nunez, P., Vazquez, R., del Toro, J. C., Bandera, A., & Sandoval, F.** (2006). Feature extraction from laser scan data based on curvature estimation for mobile robotics, in *Proceedings 2006 IEEE International Conference on Robotics and Automation*, pp. 1167-1172.

[41] **Nüchter, A.** (2011). *Robotic 3D Scan Repository*. Available: http://kos.informatik.uni-osnabrueck.de/3Dscans/

[42] **Ohno, K., Nomura, T., and Tadokoro, S. (**2006). Real-Time Robot Trajectory Estimation and 3D Map Construction using 3D Camera, in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5279-5285.

[43] **Pakki, K., Chandra, B., Gu, D., and Postlethwaite, I. (**2011). Cubature Kalman Filter based Localization and Mapping, in *IFAC WC*, , pp. 2121-2125.

[44] **Pandey, G., McBride, J. R., and Eustice, R. M.** (2011). Ford campus vision and lidar data set. *International Journal of Robotics Research*.

[45] **Pandey, G., Savarese, S., McBride, J. R., and Eustice, R. M.** (2011). "Visually bootstrapped generalized ICP," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2660-2667.

[46] **Pathak, K., Birk, A., Vaskevicius, N., and Poppinga, J.** (2010). Fast Registration Based on Noisy Planes With Unknown Correspondences for 3-D Mapping. In *IEEE Transactions on Robotics*, 26(3), 424-441.

[47] **Pathak, K., Birk, A., Vaskevicius, N., Pfingsthorn, M., Schwertfeger, S.,** et al**.** (2010). Online three-dimensional SLAM by registration of large planar surface segments and closed-form pose-graph relaxation. *J. Field Robot.*, 27(1), 52-84.

[48] **Pathak, K., Vaskevicius, N., Poppinga, J., Pfingsthorn, M., Schwertfeger,** et al. (2009)., Fast 3D mapping by matching planes extracted from range sensor point-clouds, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1150-1155.

[49] **Poppinga, J., Vaskevicius, N., Birk, A., and Pathak, K.** (2008). Fast plane detection and polygonalization in noisy 3D range images, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3378-3383.

[50] **Rusinkiewicz, S. and Levoy, M.** (2001), Efficient variants of the ICP algorithm," in *Third International Conference on 3-D Digital Imaging and Modeling*, pp. 145-152.

[51] **S. Thrun, W. B., and D. Fox**, *Probabilistic Robotics*: MIT Press, 2005.

[52] **Segal, A., Haehnel, D., and Thrun, S.** (2009). Generalized-ICP. Proceedings of *Robotics: Science and Systems*, Seattle, USA.

[53] **Steder, B., Rusu, R. B., Konolige, K., and Burgard, W.** (2010), NARF: 3D Range Image Features for Object Recognition, in *Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan.

[54] **Takeuchi, E. and Tsubouchi, T.** (2006), A 3-D Scan Matching using Improved 3-D Normal Distributions Transform for Mobile Robotic Mapping, in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pp. 3068-3073.

[55] **Takubo, T., Kaminade, T., Mae, Y., Ohara, K., and Arai, T.** (2009). NDT scan matching method for high resolution grid map, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1517-1522.

[56] **Thrun, S., Koller, D., Ghahramani, Z., Durrant-Whyte, H., and A.Y., N.** (2002). Simultaneous Mapping and Localization With Sparse Extended Information Filters. Proceedings of the *Fifth International Workshop on Algorithmic Foundations of Robotics*, Nice, France.

[57] **Thrun, S., Liu, Y., Koller, D., Ng, A. Y., Ghahramani, Z.,** et al. (2003). Simultaneous Localization and Mapping with Sparse Extended Information Filters. *The International Journal of Robotics Research*, 23, 693-716.

[58] **Thrun, S., Liu, Y., Koller, D., Ng, A. Y., Ghahramani, Z.,** et al. (2004). Simultaneous Localization and Mapping with Sparse Extended Information Filters. *The International Journal of Robotics Research*, 23, 693-716.

[59] **Ulas, C. and Temeltas, H.** (2011). A 3D Scan Matching Method Based On Multi-Layered Normal Distribution Transform, presented at the *18th IFAC World Congress*.

[60] **Ulas, C. and Temeltas, H.** (2011). 3D simultaneous localization and mapping based on multi-layered normal distribution transform, presented at the *18th Saint Petersburg International Conference on Integrated Navigation Systems*, St.Petersburg, Russia.

[61] **Ulas, C. and Temeltas, H.** (2011). A fast and robust scan matching algorithm based on feature dependent sampling, in, *9th IEEE International Conference on Control and Automation (ICCA)*, pp. 124-129.

[62] **Ulas, C. and Temeltas, H.** (2011). A fast and robust scan matching algorithm based on ML-NDT and feature extraction," in *International Conference on Mechatronics and Automation (ICMA)*, pp. 1751-1756.

[63] **Ulas, C. and Temeltas, H.** (2011). HD-SLAM based on CKF and Semantic Perception in Outdoor Environments, presented at the *43rd International Symposium on Robotics*.

[64] **Ulas, C. and Temeltas, H.** (2011). Plane-feature based 3D outdoor SLAM with Gaussian filters, in *IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, pp. 13-18.

[65] **Ulas, C. and Temeltas, H.** (2011). Randomized Sigma Point Kalman Filters for Feature Based 3D SLAM, presented at the *Bar-Itzhack Memorial Symposium on Estimation, Navigation, and Spacecraft Control*.

[66] **Ulas, C. and Temeltas, H.** (2011). A Robust Feature Extraction Method and Semantic Data Association for 6D SLAM, presented at the *14th International Symposium on Robotics and Applications*.

[67] **Walter, M. R., Eustice, R. M., and Leonard, J. J.** (2007). Exactly Sparse Extended Information Filters for Feature-based SLAM. *Int. J. Rob.* Res., 26(4), 335-359.

[68] **Wang, C.-C.** (2004). *Robotics Institute*, Carnegie Mellon University: Pittsburgh, PA.

[69] **Wang, X., Wang, F., and Li, P.** (2012). Investigation on SEIF Based on SLAM in Electronic Engineering, in *Advances in Mechanical and Electronic Engineering*. vol. 177, D. Jin and S. Lin, Eds., ed: Springer Berlin Heidelberg, 2012, pp. 593-598.

[70] **Weingarten, J. and Siegwart, R.** (2005). EKF-based 3D SLAM for structured environment reconstruction, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3834-3839.

[71] **Weingarten, J. W., Gruener, G., and Siegwart, R.** (2004). Probabilistic plane fitting in 3D and an application to robotic mapping, in *IEEE International Conference on Robotics and Automation*, pp. 927-932 Vol.1.

[72] **Wulf, l. and Wagner, B.** (2003). Fast 3D Scanning Methods for Laser Measurement Systems, in *Proceedings of the International Conference on Control Systems and Computer Science*, pp. 312-317.

[73] **Wulf, O.** (2010). *Hannover, Leibniz University Campus*. Available: http://kos.informatik.uni-osnabrueck.de/3Dscans/

[74] **Wulf, O., Nüchter, A., Hertzberg, J., and Wagner, B.** (2008). Benchmarking urban six-degree-of-freedom simultaneous localization and mapping. *J. Field Robot.*, 25(3), 148-163.

[75] **Yangming, L. and Olson, E. B.** (2010). Extracting general-purpose features from LIDAR data, in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1388-1393.

[76] **Zhang, S., Xie, L., and Adams, M. D.** (2006). Feature extraction for outdoor mobile robot navigation based on a modified Gauss–Newton optimization approach. *Robotics and Autonomous Systems*, 277-287.

CURRICULUM VITAE

Name Surname: Cihan Ulaş

Place and Date of Birth: Istanbul 19.06.1982

E-Mail: cihan.ulas@tubitak.gov.tr

B.Sc.: Department of Electronics Engineering, Uludag University, Bursa

M.Sc.: Department of Electronics Engineering, Fatih University, Istanbul

Professional Experience and Rewards: He has worked as a research assistant in Fatih University for 7 years. Currently, he has been working for TÜBİTAK for almost a year.

List of Publications and Patents:

- O. Toker, H. Gümüşkaya, C. Ulaş, B. T. Yılmaz, "Lightweight Wireless Protocol Based on IEEE 802.11 for Delay Sensitive Telerobotic Systems", *Turkish Journal of Electrıcal Engineering and Computer Sciences*, 2012.
- Cihan Ulaş, Onur Toker, Kemal Fidanboylu, "Rotation Angle Estimation Algorithms for Textures and Their Real-Time Implementation" *Machine Vision*, Intech-Open Access Publisher DOI: 10.5772/26657. 2012.
- Cihan Ulaş and H. Seçkin Efendioğlu, Onur Toker, Halûk Gümüşkaya, "Delay Sensitive Wireless Protocols for Telerobotics Applications", *Journal of Networking Technology*, Vol. 1, No. 3, pp. 118-125, Sep. 2010.
- Cihan Ulaş, H. S. Efendioğlu, O. Toker, and H. Gümüşkaya, "Delay Sensitive Wireless Protocols for Telerobotics Applications", *ICADIWT*, ISTANBUL, Aug. 2010.
- O. Toker, C. Ulaş, H. S. Efendioğlu, and H. Gümüşkaya, "Embedded Systems for Delay Sensitive Wireless Protocol Development and Analysis for Telerobotics Applications", *Int. Conf. on Wireless Networks*, Las Vegas, NV, U.S.A, Jul. 2009.
- Serdar YILMAZ, Onur TOKER, Nurullah ARSLAN, Herman SEDEF, Cihan ULAŞ, "PID ve P-Dinamik Değişim Sınırlayıcı (P-DDS) ile Debi Kontrolü", Türkiye *Otomatik Kontrol Ulusal Toplantısı*, İstanbul, Oct. 2009.
- Cihan Ulaş, Sümeyra Demir, Onur Toker, Kemal Fidanboylu, "Rotation Angle Estimation Algorithms for Textures and Their Real-Time Implementation on the FU-Smart", *ISPA*, Istanbul/Turkey, Sep. 2007.

PUBLICATIONS/PRESENTATIONS ON THE THESIS

- C. Ulas and H. Temeltas, "3D Multi-Layered Normal Distribution Transform for Fast and Long Range Scan Matching," *Journal of Intelligent & Robotic Systems*, pp. 1-24, 2012.

- C. Ulaş and H. Temeltaş, " Feature Based 3D Outdoor SLAM with Local Filters," *International Journal of Robotics and Automation*, 2012, *accepted for publication*.

- C. Ulas and H. Temeltas, "A fast and robust scan matching algorithm based on ML-NDT and feature extraction*", in International Conference on Mechatronics and Automation (ICMA)*, 2011, pp. 1751-1756.

- C. Ulas and H. Temeltas, "A 3D Scan Matching Method Based On Multi-Layered Normal Distribution Transform," *presented at the 18th IFAC World Congress*, 2011.

- C. Ulas and H. Temeltas, "3D simultaneous localization and mapping based on multi-layered normal distribution transform," *presented at the 18th Saint Petersburg International Conference on Integrated Navigation Systems*, St.Petersburg, Russia, 2011.

- C. Ulas and H. Temeltas, "A fast and robust scan matching algorithm based on feature dependent sampling," *in 9th IEEE International Conference on Control and Automation (ICCA)*, 2011, pp. 124-129.

- C. Ulas and H. Temeltas, "Plane-feature based 3D outdoor SLAM with Gaussian filters," in, *2012 IEEE International Conference on Vehicular Electronics and Safety (ICVES), 2012*, pp. 13-18.

- C. Ulas and H. Temeltas, "HD-SLAM based on CKF and Semantic Perception in Outdoor Environments," *presented at the 43rd International Symposium on Robotics*, 2012.

- C. Ulas and H. Temeltas, "A Robust Feature Extraction Method and Semantic Data Association for 6D SLAM," *presented at the 14th International Symposium on Robotics and Applications*, 2012.

- C. Ulas and H. Temeltas, "Randomized Sigma Point Kalman Filters for Feature Based 3D SLAM," *presented at the Bar-Itzhack Memorial Symposium on Estimation, Navigation*, and Spacecraft Control, 2012.

- C. Ulas, I. F. Saglar, and H. Temeltas, "3D Fast-SLAM Based on Plane Features in Outdoor Environments," *presented at the* Symposium SET-168 on Navigation Sensors And Systems in GNSS Denied Environments, 2012.