ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE
ENGINEERING AND TECHNOLOGY

DESIGN, IMPLEMENTATION AND COMPARISON OF SENSOR FUSION METHODS FOR OBJECT DETECTION AND TRACKING BASED ON MULTIPLE 3D LIDAR SENSORS

M.Sc. THESIS

Elif AKSU TASDELEN

Department of Control and Automation Engineering

Control and Automation Engineering Programme

JUNE 2020

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE
ENGINEERING AND TECHNOLOGY

DESIGN, IMPLEMENTATION AND COMPARISON OF SENSOR FUSION METHODS FOR OBJECT DETECTION AND TRACKING BASED ON MULTIPLE 3D LIDAR SENSORS

M.Sc. THESIS

Elif AKSU TASDELEN
(504161136)

Department of Control and Automation Engineering

Control and Automation Engineering Programme

Thesis Advisor: Assoc. Prof. Volkan SEZER

JUNE 2020

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ÇOKLU 3D LİDAR SENSORLERİ ÜZERİNDE NESNE ALGILAMA VE TAKİBİ İÇİN SENSÖR FÜZYON YÖNTEMLERİNİN TASARIMI, UYGULAMASI VE KARŞILAŞTIRILMASI

YÜKSEK LİSANS TEZİ

Elif AKSU TASDELEN
(504161136)

Kontrol ve Otomasyon Mühendisliği Anabilim Dalı

Kontrol ve Otomasyon Mühendisliği Programı

Tez Danışmanı: Assoc. Prof. Volkan SEZER

JUNE 2020

Elif AKSU TASDELEN, a M.Sc. student of ITU Graduate School of Science Engineering and Technology student ID 504161136 successfully defended the thesis entitled "DESIGN, IMPLEMENTATION AND COMPARISON OF SENSOR FUSION METHODS FOR OBJECT DETECTION AND TRACKING BASED ON MULTIPLE 3D LIDAR SENSORS", which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Volkan SEZER** Istanbul Technical University	....
Jury Members :	**Prof. Dr. Müjde GÜZELKAYA** Istanbul Technical University	....
	Doç. Dr. Mehmet Selçuk ARSLAN Yıldız Technical University	....

Date of Submission : **15 June 2020**
Date of Defense : **14 July 2020**

To my mom,

FOREWORD

I would like to thank my supervisor from the University, Volkan Sezer, for his advices and continuous support during my study. It was a great pleasure for me to work with him in my research.

This thesis has been studied as a part of the PRSYTINE project. I would like to thank all national funding authorities and the ECSEL Joint Undertaking, which funded the PRYSTINE project under the grant agreement number 783190. I extend my sincere thanks to Ford-Otosan for the opportunity to work on my research.

I must also thank my husband, Muhsin Taşdelen, for great support and patience. This thesis would never have been completed without his encouragement and moral support.

During my thesis study, I have benefited from the guidance and generosity of my friends. First, I would like to thank Ilhan Yunus, for spending his valuable time for my thesis, and broadening my horizon with his valuable comments and advices. I am also indebted to Mertcan Cibooğlu for his kind help, constructive criticism and support.

My deepest gratitude goes to my mother, Saniye Aksu who believed me and supported me at every moment of my life. She always tried to do her best for me. She reminds me of what life is about even if she passed away. I also owe thanks to my beloved family. I am indebted to my father, Necati Aksu, my sister, Tuba Aksu and my brother, Yusuf Aksu, not only for their endless love and support in my life, but also their patience.

June 2020 Elif AKSU TASDELEN

TABLE OF CONTENTS

Page

ABBREVIATIONS

ADAS	: Advanced Driver Assistant Systems
CTRVM	: Constant Turn-Rate and Velocity Model
CVM	: Constant Velocity Model
DBSCAN	: Density-based Spatial Clustering
DTAMR	: Detection and Tracking After Merge Raw-data
EKF	: Extended Kalman Filter
GNN	: Global Nearest Neighbour
IMM	: Interacting Multiple Model
LKF	: Linear Kalman Filter
MSE	: Mean Square Error
ODT	: Object Detection and Tracking
PDAF	: Probabilistic Data Association Filter
RANSAC	: Random Sample Consensus
RMM	: Random Motion Model
ROS	: Robot Operating System
SD	: Standard Deviation
T2TFML	: Track-to-track Fusion of Multiple LIDAR
UKF	: Unscented Kalman Filter

SYMBOLS

$\mathbf{X_t}$,	: Object State Matrix at time t
$\mathbf{Q_t}$	: Model Error Covariance Matrix
$\mathbf{A_t}$	: Transition Matrix
$\mathbf{P_t}$	: State Covariance Matrix
$\mathbf{S_t}$	: Innovation Covariance Matrix
$\mathbf{K_t}$	: Kalman gain
$\mu_{\mathbf{j}}$	: Mode probability of j model
$\mu_{\mathbf{j\|i}}$	: Conditional mode probability of transition of i^{th} model to j^{th} model
x, y	: Relative positions
v	: Relative velocity
t	: Time
ϕ	: Yaw angle
$\dot{\phi}$	: Yaw rate
$\hat{X}$	: State prediction
β	: Association probability
$\mathbf{V}$, $\mathbf{v_i}$	: Innovation

LIST OF TABLES

Page

LIST OF FIGURES

Page

DESIGN, IMPLEMENTATION AND COMPARISON OF SENSOR FUSION METHODS FOR OBJECT DETECTION AND TRACKING BASED ON MULTIPLE 3D LIDAR SENSORS

SUMMARY

The automotive industry is a significant social and financial power around the world. This industry is constantly changing with the technological developments to meet customer needs such as better performance, safety driving etc. Today, many automotive companies are bringing their vehicles to a certain level of autonomy with Advanced Driver Assistant Systems (ADAS) to provide safe and comfortable vehicles. The increasing demand in this area and the investments of automotive companies make the subject of autonomous driving more attractive. In order to increase the autonomy of the vehicle, human intervention from the system should be removed and the vehicle must do everything which the driver does. Thus, researchers and automotive companies focus on completely eliminating human intervention from these autonomous systems.

Autonomous systems require a complete perception of the surroundings and understanding the behaviour of objects around the vehicle. Accordingly, it is extremely important to detecting and tracking the objects on the environment in order to decide the motion of the vehicle.

The reliable, verified environmental perception is provided by the use of multiple sensors with the correct combination. Information coming from multiple sensors is combined with two different strategies in literature; low level fusion and high level fusion. The low level fusion strategy is used for fusing unprocessed raw data of the sensors. The merged raw data is examined for object detection and tracking. On the other hand, the high level fusion strategy first collects predictions about objects from the sensors and then, sends the tracked objects to the fusion algorithm.

Our study is focused on the usage of multiple 3D LIDAR sensors for detecting and tracking objects in real time.

The first contribution of this work is proposing and implementing "high level track-to-track fusion" method on multiple 3D LIDAR sensors. In the presented high level fusion, tracked objects from each LIDAR sensor are combined with Covariance Intersection fusion method. To the best of our knowledge, this is the first automotive application of track-to-track fusion method on multiple 3D LIDARs.

Within the research we have also applied and tested well known and studied low level fusion of multiple 3D LIDARs. In the low-level fusion approach, the raw point cloud data from each LIDAR sensor is directly fused and object detection and tracking algorithms are applied on this fused data.

These two real-time fusion strategies are implemented in the experimental test truck which is instrumented with two 3D LIDAR sensors and the performance of the fusion strategies are tested under three different driving scenarios. Additionally, the ground

truth data is collected with the help of global navigation satellite system (GNSS) in high accuracy for performance evaluation. The test results are analyzed in terms of defined performance criteria and the benefits & weaknesses of the proposed approach are discussed in this work.

To sum up in this thesis, the first part gives the overall picture of the thesis aim. Then, literature review is summarized. Afterwards, the object detection and tracking strategies will introduce at the Chapter-3. In Chapter-4, the fusion approaches will be summarized with their mathematical backgrounds. The proposed high-level fusion methodology will present in this Chapter. Chapter-5 will cover the experimental setup to test the fusion approaches' results. Accordingly, the real test results of these two approaches through the scenarios and the comparisons of the approaches will discussed in Section-6. Finally, the conclusion and planning future works on these topics will represent at the last part.

**ÇOKLU 3D LİDAR SENSORLERİ ÜZERİNDE
NESNE ALGILAMA VE TAKİBİ İÇİN SENSÖR FÜZYON YÖNTEMLERİNİN
TASARIMI, UYGULAMASI VE KARŞILAŞTIRILMASI**

ÖZET

Otomotiv endüstrisi, sosyal ve finansal açıdan dünya çapında önemli bir güçtür. Bu endüstri, müşteri ihtiyaçlarını karşılamak için emniyetli sürüş ve daha iyi performans gibi teknolojik gelişmelerle birlikte sürekli değişmektedir. Günümüzde, müşterinin artan talebiyle, birçok otomotiv şirketi güvenli ve rahat bir sürüş sağlamak için Gelişmiş Sürücü Asistan Sistemleri (ADAS) ile araçlarını belirli bir otonom seviyesine getirmektedir. Bu alanda artan talep ve otomotiv şirketlerinin yatırımları ise otonom sürüş konusunu daha cazip hale getirmiştir.

Aracın otonom özelliklerini artırmak için sistemden insan müdahalesi kaldırılmalı ve araç sürücünün yaptığı her şeyi yapabilmelidir. Bu nedenle, bu alandaki araştırmacıların ve otomotiv şirketlerinin odak çalışmaları araç sistemlerinden insan müdahalesini tamamen ortadan kaldırmak üzerinedir. İnsan etkisini kaldırmanın en büyük adımı ise insanın algıladığı kadar aracın da çevresini algılamasını ve anlamlandırmasını sağlamaktır. Aracın kendini sağlıklı bir şekilde kontrol edebilmesi ve uygun hamleleri yapabilmesi için çevresindeki diğer araç ve insanların hareketlerini/ durumlarını algılamalı ve bunları takip edip gelecek durumları için tahminde bulunmalıdır.

Bahsettiğimiz bu çevre algılama, sensörler ile mümkündür. Günümüzde otomotiv endüstrisi için üretilen ve otonom araç çalışmalarında kullanılan birçok çeşit sensör bulunmaktadır. Aracın etrafındaki nesneleri algılamak için kullanılan sensörlerden başlıcaları; Kamera, Radar ve LIDAR sensörleridir. Aracın üzerine farklı konumlarda bu gibi sensörler yerleştirerek, aracın etrafından bilgi toplanması sağlanır. Güvenilir ve doğrulanmış çevresel algı, genellikle birden fazla sensörün birlikte kullanılmasıyla oluşur. Bunun nedeni, hiç bir sensörün tek başına yeterli ve kesin bilgi verememesidir. Her sensörün belirli bir görüş alanı bulunur ve her sensör çevre hakkında farklı bilgiler sağlar. Örneğin, kamera sensörü 2 boyutlu görüntüyü bilgi olarak verirken bu görüntü hakkında uzaklık bilgisi veremez. Diğer yandan, radar sensörü mesafe bilgisini radyo sinyalleri kullanarak sağlayabilir ama objeyi sınıflandırma konusunda zayıf kalır ve yan yana olan birkaç nesneyi birbirinden ayırması zordur. Ne var ki, Radar'ın kullanım alanı çok geniştir, zorlu hava koşullarında bile verimli çalışabilir. LIDAR sensörü ise lazer ışınlarını 360 derece etrafa gönderip toplayarak yüksek çözünürlüklü ve yoğun bilgi sağlamaktadır. Lidar çok iyi bir şekilde yüzey analizi yapabildiği için nesnelerin şekillerini algılama ve bu şekilleri birbirinden ayırma yeteneğine çok yüksek düzeyde sahiptir. LIDAR sensörünün dezavantajı olarak da olumsuz hava koşullarından etkilenmesi gösterilebilir. Bu çeşitli sensörlerin tek bir çevre bilgisini oluşturabilmesi için doğru bir kombinasyonla ve yorumlamayla birleştirilmesi gerekmektedir. Böylelikle her bir sensörün avantajlı yanı birlikte kullanılabilir.

Birden fazla sensörden gelen bilgiler, literatürde iki farklı strateji ile birleştirilir. Bunlar; düşük seviyeli veri füzyonu ve yüksek seviyeli veri füzyonudur. Bu sensör füzyon yöntemleri, sensör verilerini hangi seviyede kullandığına göre değişmektedir. Eğer sensör verisi hiç işlenilmemiş ya da sadece ön bir işlemden geçirilip belli özellikleri çıkartılmışsa bu veriye düşük seviye sensör verisi denir. Düşük seviyeli füzyon stratejisi, sensörlerin işlenmemiş ham verilerinin birleştirilmesine dayanmaktadır. Birleştirilen ham veriler, nesne algılama ve takip işlemine girer. Öte yandan, sensör verileri işlenerek obje bilgileri elde edilirse ve kestirim yöntemleriyle tahminleri çıkarılırsa bu sensör çıktısı yüksek seviyede sensör verisi olarak adlandırılır. Yüksek seviyeli füzyon stratejisi önce sensörlerden nesneler hakkında tahminler toplar ve ardından nesnelerin kestirimlerini füzyon metodu kullanarak birleştirir.

Biz de bu çalışmada, aracın çevresindeki nesneleri gerçek zamanlı olarak tespit etmek ve izlemek için çoklu 3D LIDAR sensörlerinin kullanımına odaklandık. Çoklu LIDAR sensörlerini birleştirmedeki amaç, daha kesin obje tespitleri ve tahminleri elde etmektir. Çevre algısı için çoklu 3D LIDAR sensörlerde iki çeşit (düşük seviye ve yüksek seviye) gerçek zamanlı sensör füzyon yaklaşımı uygulanıp test edilmiştir. İlk olarak düşük seviyeli füzyon yaklaşımında, her bir LIDAR sensöründen alınan ham nokta bulutu verileri doğrudan birleştirilir ve bu birleştirilmiş verilere nesne algılama ve izleme algoritmaları uygulanır. Literatürde yer alan çalışmalara göre bu yöntem 3D LIDAR sensörleri için çeşitli uygulamalarda kullanılmıştır. Tez çalışmasında da ilk olarak bu yöntem uygulanmıştır.

Bu tez çalışmasının ilk katkısı, çoklu 3D LIDAR sensörlerininin verilerini birleştirmek için “yüksek seviyede obje tahminlerinin füzyonu” yaklaşımını önermek ve sensörler üzerinde bu yaklaşımı uygulamaktır. Sunulan yüksek düzey füzyonda, her bir LIDAR sensöründen nesne tespitleri ve bu nesnelere ait kestirimler toplanır ve bu kestirimler literaturde geçen ‘Covariance Intersection’ füzyon yöntemiyle birleştirilir.

İki füzyon yöntemi de aynı nesne tespit ve takip algoritmalarını kullanmaktadır. Kullanılan nesne tespit algoritması 3D LIDAR sensörünün nokta bulutu şeklinde verisinden nesnelerin çıkarılmasını sağlamaktadır. Genel olarak üç aşamada nesne verisi çıkartılır. İlk olarak yere ait olan noktalar tespit edilir ve eğim bazlı bir algoritma ile noktalar arasındaki yükseklik farkına bakılır. Yükseklik farkı az olan noktalar yer noktası olarak işaretlenerek nokta bulutundan çıkarılır. İkinci olarak, kalan nokta bulutu içindeki noktalar incelenir ve bir yakınlık tanımı yapılarak, birbirine yakın olan noktalar gruplandırılır. Son olarak da bu grupları ifade etmek için etrafına kutular çizilir ve her bir kutu bir nesneyi temsil eder.

Nesne tespit algoritmasından elde edilen nesnelerin merkez noktaları Kalman filtresi tabanlı bir takip algoritmasıyla takip edilmeye başlanır. Nesnelerin önceki konumlarından ve hareket modelleri kullanılarak çıkartılan hız ve yönelim bilgilerinden şimdiki zamana ait tahminleri çıkartılır. Bu noktada önemli olan hareketleri yansıtabilecek hareket modelini seçmektir. Bu çalışmanın içerisinde üç hareket modeli aynı anda kullanılmaktadır. Bunun için, çoklu hareket modellerinin birlikte kullanılmasını sağlayan ‘Interacting Multiple Model (IMM)’ yöntemi Kalman filtresiyle birleştirilerek kullanılmıştır. Bu tahminler yeni elde edilen sensör verileriyle karşılaştırılıp doğru sensör bilgisiyle güncellenir. İki füzyon yaklaşımının da çıkışından elde edilen sonuçlar; takip edilmiş nesnenin pozisyon bilgisi, hız bilgisi, yön açısı ve yön açısın değişimi bilgileridir.

Bu iki gerçek zamanlı füzyon stratejisini test etmek için Ford F-Max tırı test aracı olarak kullanılmıştır. İki 3D LIDAR sensörü tırın ön köşelerine yerleştirilmiştir. Bu şekilde sensör yerleşimi test aracının ön tarafını her iki sensörün de ortak görüş alanı olarak elde edilmesini sağlamaktadır. Füzyon algoritmalarının sonuçları bu ortak alan içerisinde değerlendirilmektedir. Uygulanan stratejilerinin performansı üç farklı sürüş senaryosu altında test edilmiştir. Senaryolar hedef aracın yanal ve dikey pozisyonlarını ve hızını değerlendirmek için tasarlanmıştır. Ek olarak, performans değerlendirmesi için yüksek doğrulukta veriler küresel navigasyon uydu sistemi (GNSS) yardımıyla toplanmıştır.

Gerçek zamanlı test sonuçları toplandıktan sonra bu sonuçları değerlendirmek için üç performans kriteri tanımlanmıştır. İlk olarak bu algoritmaların işlem sürelerine bakılmıştır. Süreler incelendiğinde aralarında farklılık olsa da iki füzyon algoritmasının da işlem süresi LIDAR sensörünün gönderme süre aralığından az çıkmaktadır. Diğer iki performans metrikleri ise Ortalama Kare Hatası (MSE) ve Standart Sapma (SD)'dır. Ortalama Kare Hatası nesne tahminlerinin doğruluğu hakkında bilgi sağlar. Standart sapma ise sonuçların gerçek verilere göre dağılımını gösterir. Senorya sonuçlarından elde edilen tahminler gerçek verilerle karşılaştırılarak bu değerler elde edilir. Önerilen yaklaşımın yararları ve zayıflıkları tanımlanan performans kriterlerine göre analiz edilerek tartışılmıştır. Sonuçlar incelendiğinde önerilen yeni yöntemin nesne takibindeki hataları iyileştirdiği görülmüştür.

Özetlemek gerekirse bu tezde; ilk bölüm, tez amacını ve çalışmanın genel resmini vermektedir. Daha sonra, Bölüm-3'te nesne tespit ve takip stratejileri tanıtılacaktır. Bölüm-4'te seçilen obje tespit ve takip mimarisi açıklanarak füzyon yaklaşımları matematiksel dayanakları ile özetlenecektir. Ayrıca bu bölümde önerilen üst düzey füzyon metodolojisi de sunulacaktır. Bölüm-5, füzyon yaklaşımlarının sonuçlarını test etmek için deney düzeneğini ele almaktadır. Bölüm-6'da ise tasarlanan senaryolardaki bu iki yaklaşımın test sonuçları ve yaklaşımların gerçek verilerle karşılaştırılması bulunmaktadır. Son olarak, sonuçlar kısmında, elde edilen sonuçlar ve bu sonuçlara ilişkin gelecekteki çalışmaların planlanması tartışılacaktır.

1. INTRODUCTION

The automotive industry is a significant social and financial power around the world. This industry is constantly changing with the technological developments to meet customer needs such as better performance, safety driving etc. Today, many automotive companies are bringing their vehicles to a certain level of autonomy with Advanced Driver Assistant Systems (ADAS) to provide safe and comfortable vehicles. The increasing demand in this area and the investments of automotive companies make the subject of autonomous driving more attractive. Autonomous driving means that the vehicle can go anywhere without human intervention, in general.

Autonomous driving is defined by the The Society of Automotive Engineer (SAE) Internationals as six levels, depending on their usage and the degree of human participation. Figure 1.1 shows the determined autonomous levels. As can be

Figure 1.1: SAE J3016 Levels of Driving Automation [2].

understood from the figure, as the level increases, the vehicle's level of autonomy increases. For example, Level 0 vehicles are the traditional vehicles which manually controlled by driver. On the other hand, Level 5 represents full automation vehicles which do not require driver. According to this classification, commercially available systems such as Adaptive Cruise Control, Park Assists are part of autonomous systems to some extent. In order to increase the autonomy of the vehicle, human intervention from the system should be removed and the vehicle must do everything which the driver does. Thus, researchers and automotive companies focus on completely eliminating human intervention from these autonomous systems. This requires a complete perception of the surroundings and understanding the behavior of objects around the vehicle. Accordingly, it is extremely important to detecting and tracking the objects on the environment in order to understand the behavior of objects and to decide the motion of the vehicle. The solution for this perception problem is directly related to the developments on sensor technology.

In autonomous vehicle studies, Radar and Camera are the most used sensors in recent years. The camera sensor provides vision like the human eye, but gives 2D images from its field of view. The 3D camera sensors are developed to display detailed information about the images with extra depth information. The Radar (Radio Detection and Range) sensor sends radio waves and calculates the distance and speed from the return waves. Thus, radar sensor provides position and speed information of objects by the Doppler shift effect. Since the radar and camera sensors complement each other, it is generally attempted to make environmental perception with their fusion. However, some drawbacks still remain such as the sunlight effects, limited field of view etc.

Recently, the Light Detector and Range (LIDAR) sensor offers an alternative solution that has been used for a long time in the defense industry.The main idea under this sensor is to send laser beams and measure time of flights from the returned beams with the same principle of operation of the radar sensor. LIDAR sensors can be 1D, 2D and 3D, depending on scanning technology and number of beams. In autonomous vehicles, spinning 3D LIDAR sensors are mostly used due to the its 360° field of view with high resolution. The 3D properties of objects can be achieved with the advantage of high-resolution data, and the LIDAR sensor operates both day and night. These 3D Lidar sensors are widely used as mounting on the top of the vehicle LIDAR sensor fusion has been studied by creating different combinations with other sensors. Since, reliable environmental perception is provided by the use of multiple sensors and the correct combination of these sensors. The main focus of this combination is to obtain reliable, stable and verified information from several sensors. Different fusion strategies have been studied for different data abstraction of the original sensor data and for different sensor types. Generally, fusion strategies are gathered under two categories; low level fusion and high level fusion [7]. The low level fusion strategy is used for fusing unprocessed raw data of the sensors. The huge raw data is examined for object detection and tracking. On the other hand, the high level fusion strategy first collects predictions about objects from the sensors and then, sends the tracked objects to the fusion algorithm. At last step of high level fusion method, sensor tracked objects need to be combined. In high level fusion strategy, tracking outputs from the sensors are considered as measurements of the fusion algorithm. These fusion strategies construct the main focus of the thesis.

This thesis are involved autonomous track developments of Ford Otosan and funded by European Project, Prystine.

2. LITERATURE REVIEW

In literature, LIDAR data fusion has been studied with different type of sensors such as radar, camera or 2D Lidar scanners [8]. In [8], LiDAR and camera sensors' raw data are combined after synchronization. 3D LiDAR point cloud data is projected to the camera image and then, the colored point cloud is used for object detection. The paper [9] proposes combining Lidar-Camera data with their complementary features. This fusion also starts with projection of point cloud data into the camera image. These papers use the raw data fusion strategy. In the 2007 DARPA Urban Challenge, the Stanford team also proposes low level fusion of 3D Lidar sensor with six 2D lidar scanners for object detection and tracking [10]. In [11] one 3D Lidar and five 2D Lidar scanners are processed in real time. Most of the studies include single 3D lidar sensor and rely on fusing it with the other sensors.

Our study is focused on the usage of multiple 3D LIDAR sensors for Object Detection and Tracking (ODT) in real time. The ODT based on multiple 3D LIDARs has been studied in [12] which point cloud data from each LIDAR sensor were merged and then applied ODT. Similarly, [13] uses the same method only for the pedestrian detection. These proposed [12] and [13] methods are called low level fusion since raw point cloud data is directly combined. On the other hand, track to track fusion strategy is another common approach for multiple sensor fusion. The tracking results are get from single 3D LiDAR sensor and these tracks are fused with other sensors' tracks. An example of track-to-track fusion which fuses 3D LIDAR and Radar sensors' tracked objects can be found in [14].

However, considering the papers that have been published so far, none of the authors have proposed fusing multiple 3D LIDAR sensors with track-to-track fusion strategy. The first contribution of this work is proposing and implementing "high level track-to-track fusion" method on multiple 3D LIDAR sensors.

The other contribution is the analysis and comparison of track-to-track fusion performance with the well-studied low-level fusion method. For this aim, both low level fusion and the track-to-track fusion approaches are implemented using multiple 3D LIDAR sensors for object detection and tracking. The tracking results of both methods are analyzed and their performances are evaluated using the ground truth data. Experiments are performed in proving ground using a heavy duty test truck which is equipped with multiple 3D LIDAR sensors. Three different driving scenarios are used to evaluate implemented two fusion approaches.

The thesis is organized as follows. Until this chapter, a brief overview of autonomous vehicle levels and autonomous vehicle sensors are given and the purpose of the thesis is introduced. In the next section, the Lidar sensor and the object detection and tracking processes of LIDAR sensor that are frequently used in the literature will be explained in detail. In Chapter 3, selected object detection and tracking architecture will be summarized and the technical approaches to combine more than one Lidar will be explained. Accordingly, these approaches and their results regarding the test environment will be discussed in Chapter 4. Finally, the conclusion and future work will be in the final chapter.

3. MULTIPLE-OBJECT DETECTION AND TRACKING

This chapter aims to give an overview of the object detection and tracking processes. The detection problem is defined as the finding objects which are in the field of view of the sensor. The objects in the environment can be moving objects such as cars, pedestrians or standing objects such as buildings and trees. It is important to decide which objects will be detected with the object detection algorithm. In this thesis, pedestrians and cars are the main focus of the detection process.

Object detection occurs each scan and does not include the concept of time and continuity. Thus, when detecting an object, these algorithms treat it as a new object each time. The tracking process is necessary to understand the history of the objects. Since, if the object is tracked, the future position of the object can be predicted based on the current position and the velocity which is computed from previous scans. As a result, object tracking algorithms provide the understanding the behavior of the detected objects. These two process which are called as Object Detection and Tracking (ODT) in this study are generally implemented together and the detected objects are fed to the tracking process in each cycle. Generalized workflow of the ODT is represented in the following figure.

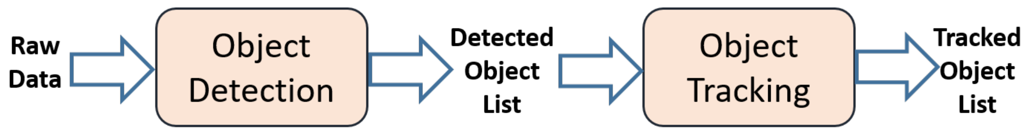

Figure 3.1: Generalized workflow of the ODT.

The detection algorithm is customized for each sensor according to its data. In order to implement proper algorithms, it is important to understand LIDAR sensor data. Therefore, technical details of the LIDAR sensor will be given in the next section. Then, detection and tracking processes which are used on LIDAR data in literature will be explained.

3.1 3D LIDAR Scanner

LIDAR sensor is an active remote laser scanning sensor. The basic operating concept is the same as the radar but the difference is that LIDAR uses light beams instead of radio waves. LIDAR sensor basically generates the laser beams and transmits them. Then, the travel time of returned beam is measured to calculate the distance of the object [15]. The distance calculation of the object is done by using the speed of light as in (3.1). Figure 3.2 represents basically the LIDAR pulses.

$$Distance = (\text{Speed of light} * \text{Travel time})/2 \tag{3.1}$$

These laser beams are sent from transmitter and received by receiver. There are

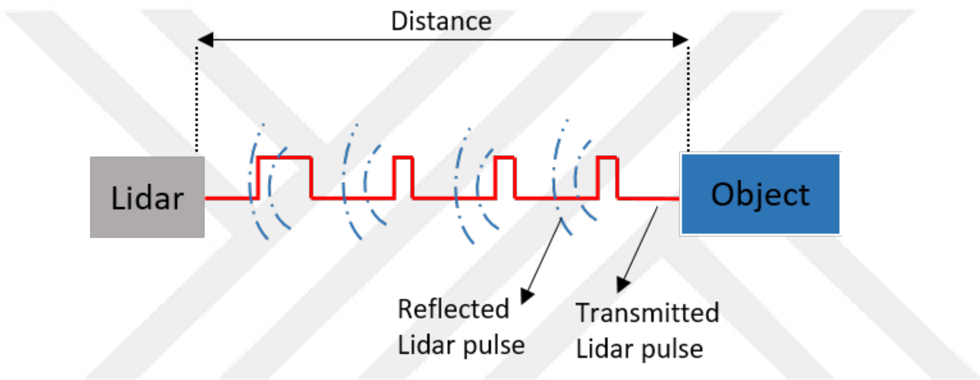

Figure 3.2: LIDAR pulses and distance.

two main components in LIDAR systems; transmitter and receiver. When these two components are placed in the center and rotated around the center, this system measures around 360 degrees. Spinning 3D LIDAR sensors work based on this principle. Most modern 3D LIDAR scanners use a rotating mechanism and mirrors to get point cloud data from 360 degree of environment.

Velodyne is the popular LIDAR technology company which provides LIDAR solutions for autonomous vehicles. Also, Velodyne 3D LIDAR sensors are available product on the market. Accordingly, most of the autonomous vehicles studies have been done with using Velodyne's 3D LIDAR sensors as a reference sensor [16]. In this thesis, Velodyne VLP-16 LIDAR sensors are used. The 3D Scanner rotation mechanism and the image of VLP-16 LIDAR sensor are given in the Figure 3.3. Velodyne VLP-16 LIDAR sensor includes 16 transmitter and receiver pairs on the vertical direction and and measures 360° of surround with its rotating mechanism. In the vertical direction, there is 2° between each beam so its vertical field of view is 30°. Rotation frequency

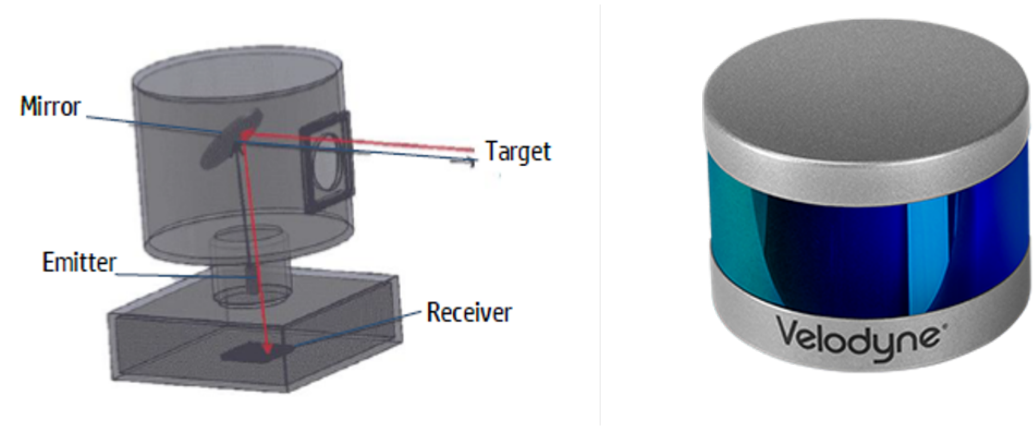

Figure 3.3: 3D LIDAR mechanism [3] and Velodyne VLP-16 3D LIDAR Sensor [4].

is adjustable between 5 Hz and 20 Hz. At the maximum rate VLP-16 LIDAR gives 300.000 points per second. The points are returned up to 100 meters [1]. The specifications of this LIDAR are given in the Table 3.1. The VLP-16 LIDAR outputs

Table 3.1: VLP-16 Specifications [1]

Attributes	**Value**
Channel Number	16
Vertical FOV	30°
Horizontal FOV	360°
Angular Resolution	0.1° - 0.4°
Measurement Range	100m
Range Accuracy	3cm
Rotation Rate	5Hz - 20 Hz

are get via Ethernet. Ethernet packets are parsed with using driver packages. The output of the VLP-16 driver is represented with the Point Cloud Data which includes x,y,z positions and the intensity of the points. The visualization of an example point cloud data is given in the Figure 3.4. The color of the points changes according to the intensity. Intensity comes from texture applied to material which laser beam hits. When laser beam hits the texture it returns RGB color format then, converts this RGB value to gray-scale (0..255) and uses this number as intensity. The number of point in the cloud is extremely high and it makes the data processing very challenging. There are many multi object detection algorithms proposed in the literature for point cloud data. The algorithms in this paper involves two successive steps; object detection and tracking as explained above. The details will be given in the following sections.

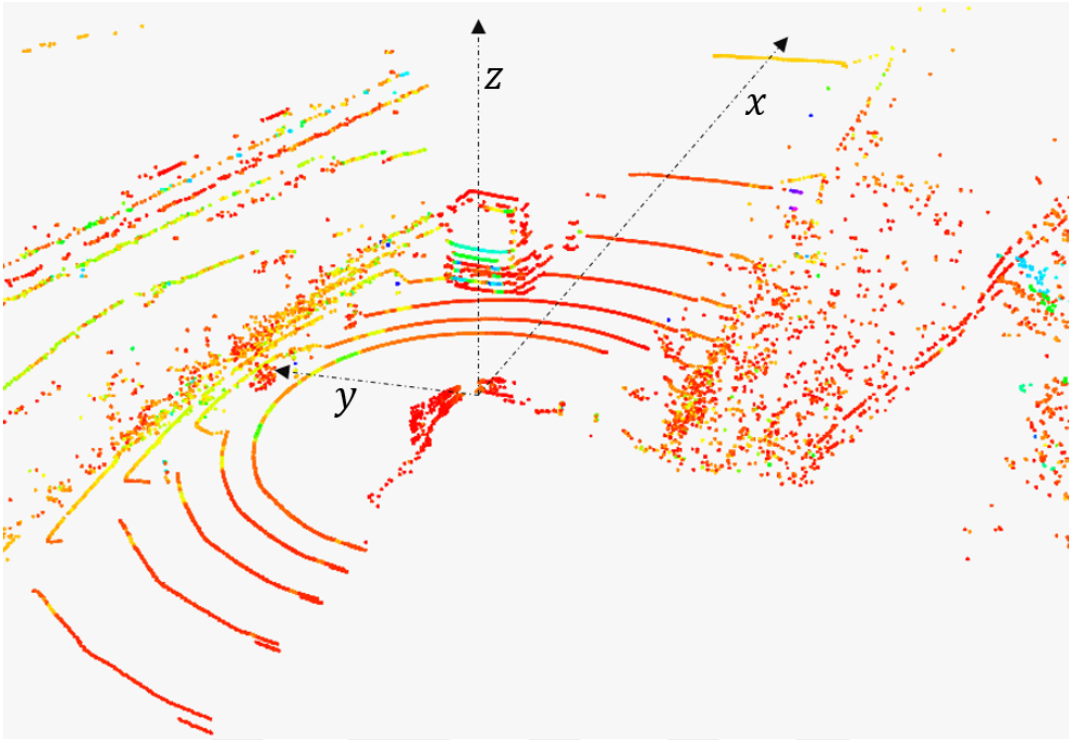

Figure 3.4: Point cloud data.

3.2 3D LIDAR Multi-Object Detection

In this section, the object detection techniques and the used detection strategies for LIDAR sensor are presented. Getting object information from point cloud data has very common steps; ground extraction, point cloud clustering and the bounding box fitting. Figure 3.5 provides an overview of the steps followed in point cloud processing for object detection.

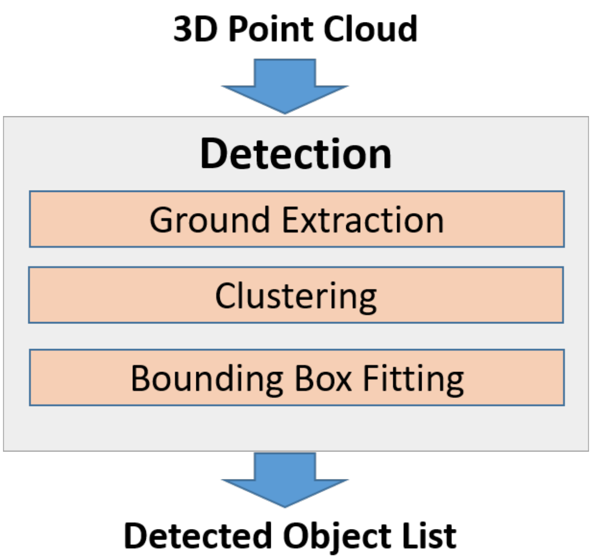

Figure 3.5: Detection process overview.

3.2.1 Ground removal

The first step in the detection algorithm is to separate 3D point cloud data into ground points and non-ground points. Then, the founded ground points are extracted from the point cloud data. The main idea of the extraction is that working on ground points is unnecessary to find the objects. The ground point extraction helps to decrease computational work with decreasing the point number.

The ground points can be found using different geometrical approaches. The first approach is the height threshold method [17] which assumes that ground points are located under some height value. Point's height from the ground is calculated using the known location of the sensor. This height value of the point is compared with the threshold if it is lower than the threshold, point is removed from data-set. This method very useful to find ground clearly if the exact location of the LIDAR is known. However, any movement or orientation on the LIDAR can cause a problem.

The most of the remaining methods starts by projecting points into the grid cells. Generally, 2D grid maps which include x, y information are generated. If the height information is added, these grids build 2.5D elevation map. The underlying reason for this projection is that grid cells reduce the run-time of the search algorithms. The common methods searching ground points in this way are: slope-based methods, scan-based methods and surface-based methods. In the slope based method, the height differences between the cells are computed [18]. The key point of this method is that the height difference of the cells will be small, if the points are belong to ground surface. The surface based methods are iteratively taking into account the distance and the angle of the cells as well as the heights of them [19]. A surface model is used to define ground and points which are above this surface become non-ground points. The most important part of this method is the selecting surface model. The scan-based methods attempt to fit a plane as a ground and RANSAC based algorithms are used for the accurate plane extraction.

On the other hand, learning based methods are also used for finding ground points. However, these algorithms need huge data-set for training.

3.2.2 Point cloud clustering

The next step of the object detection algorithm is the investigating non-ground points to find the object clusters. The clustering means that group the points which are close to each other. The reason for using the proximity is that the points of the same object are expected to be close to each other as a location. There are many ways how to do this grouping, but only the four most used clustering methods for point cloud data will be explained closely. The first grouping method is the hierarchical based clustering that organized clusters as a tree. These tree can be constructed with two ways; bottom-up (Agglomerative) or top-down (Divisive). In the bottom-up approach, grouping starts with a point / cluster and is combined by looking at the distance of the nearby clusters until it is above the threshold. The top-down approach starts with a large cluster containing every clusters, and then divides this cluster into small clusters [20]. The agglomerative approach between these two hierarchical clustering methods is the most common way for LIDAR point cloud data clustering [21].

In order to implement clustering algorithm, proximity between data points must be defined. The different algorithms choose different proximity definition. Euclidean distance is often used to measure the proximity between points. The hierarchical clustering starts with a one point or with multiple points. The distance of neighboring points with this starting point is calculated and if they are close to each other, that is, the distance is lower than the threshold value, the algorithm labels them as the same cluster. The distance threshold and linkage criteria are used to determine the number of iterations to form a cluster. The main advantage of this clustering method is the number of clusters are not determined before initializing the algorithm.

The second grouping method is centroid-based or partitioning-based clustering algorithm. The algorithm starts with the specific number of k partition with their initial center points. The closest points of these centers are assigned as the clusters. Then, the mean of clusters is computed again and clustering algorithm runs according to these new mean values. This method includes algorithms such as K-means, Fuzzy c-mean, K-moloid to do optimal clustering [22]. The final results depend on the selection of initial center points.

Moreover, Density-based clustering methods assume that clusters can be separated by considering their densities, because point cloud data is a mixture of high and low densities [23]. High density regions indicate the clusters. Density connected points are searched for every directions to increase density. This searching method only depends on the density, so, clusters in the point cloud data can have arbitrary shapes. The different shapes provides clustering the different objects such as pedestrians, cars etc. There are two popular algorithm for density based clustering; euclidean clustering and DBSCAN(density-based spatial clustering of applications with noise). These methods are successful in discovering random shaped objects and giving effective results in large databases [24]. The DBSCAN method includes two parameters which are the searching radius and minimum number of points. However, it is difficult to adjust the searching radius since different radius values give the different cluster results. Therefore, DBSCAN algorithms make a difference in finding method of this parameter [25].

The other grouping method is model based clustering which is also called distribution based clustering. Model-based algorithm tries to fit each cluster one probabilistic distribution model. [26]. Firstly, distribution parameters (mean, variance) for each cluster are initialized and then, each points probability of being a same cluster is calculated according to the distance of the distribution mean. The maximum likelihood is computed for maximize the probability of the cluster.

3.2.3 Bounding box fitting

Clustering algorithms groups the points but does not give the exact information for tracking. In order to get the object information such as volume of the object, center positions of the object etc., bounding box fitting is applied. There are three main ways to fit a box in the clusters. One of them is model based estimation which try to fit these boxes into a known geometrical model. For the vehicle detection, cuboid or rectangular models are preferred. The rectangular shape is not sufficient to define the 3D object so generally cuboid model is used. In order to find best fit cuboid shape, optimization methods or extra algorithms are used. [27] fits one cuboid box at the center of the cluster. Then, Iterative Closest Point (ICP) algorithm changes the pose and dimension of the box each iteration to find best fit. [28] proposes to fit rectangular shape because the height of the vehicle is not used on tracking. The rectangular shape is fitted on the center of cluster then position of the center changes over time. These methods finds the boxes and the center of the box with the used iteration methods.

Minimum-area-rectangular method is implemented to fit unrelated boxes. However, orientation is also needed for define the bounding box in a correct way. In order to find orientation, vehicle features are used. On the other hand, in order to fit bounding box the edge features can be used firstly. The focus of this method is to find edge of the targets from clusters and fit the box according to the edges. Edge lines are found from the dominant line on the cluster. These edges also can be a specific shape like L-shape, corners. The Figure 3.6 represents an example study of L-shape fitting. First, two points are determined from within the cluster that will form a line of maximum length. Then, the third point with maximum distance perpendicular to this point is found and L shape is obtained. These points also form the three corners of the box.

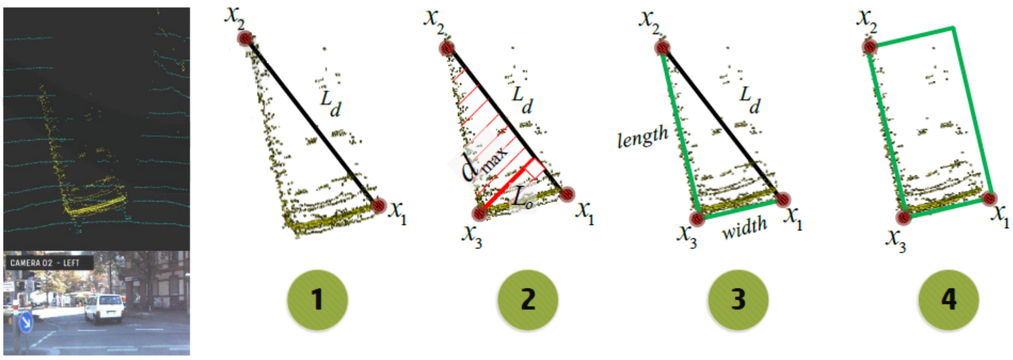

Figure 3.6: L-shape fitting example [5].

3.2.4 3D LiDAR multi-object detection architectures in literature

Multi Object Detection algorithms are combination of different solutions of these three main detection steps mentioned above. For instance, [29] uses RANSAC algorithm for fitting the plane on ground. Non-ground part is downsampled and voxel grid filtering is applied to find objects. Comparison between two point clouds on two different timestamps provides detection of dynamic objects. In [30], authors use Jump Distance Feature to find the feature segments and group them into the clusters. Bounding points of clusters are fitted into polyline. This polylines represent the bounding boxes at end of the algorithm.

In [3, 5, 12], ground is found with slope based classification and 3D point cloud is projected to occupancy grid. After that, the hierarchical Connected Component Clustering method is used for detecting objects. Finally, the combination of minimum area rectangle algorithm and L-Shape fitting method are used for feature based bounding box fitting.

3.3 Multi-Object Tracking

The multi-objects tracking algorithm is the second part of the ODT process. The object tracking cycle includes state estimation, data association and state update steps which are seen in the Figure 3.7. The state estimation computes the expected evolution of the objects' states over time, data association is to find the right sensor measurement for the tracked object and state update corrects states of the objects with the associated measurement. As can be seen from the workflows, the outputs of the detection process

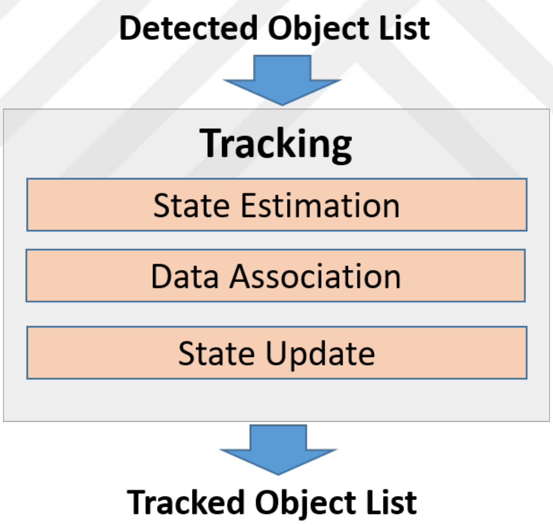

Figure 3.7: Detection process overview.

are the inputs of the tracking process. Bayesian methods are used on the detected objects' state estimation. The typical bayesian methods which are used tracking are Linear Kalman Filter (LKF), Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF). The choice of Kalman filter depends on the linearity of the motion model [31]. For multi object tracking, linear and non-linear models are usually used together. The reason for using multiple models is that the objects do not move in accordance with a specific model in the environment. Moreover, the tracked object may change its movement model during the tracking process.

Interacting Multiple Model (IMM) provides a switching mechanism between different motion models according to the sensor measurements [32]. For the advantage of the multiple models, IMM based state estimation algorithm is implemented for multi-object tracking.

After state estimation, due to the background noises and clutter in the environment, it is not easy to decide which of the measurements belong to which tracked object. For this reason, a data association is required as middle step among the state estimation and the state update. There are several approaches for associating the measurements such as; Nearest Neighborhood (NN), Probabilistic Data Association (PDA), Joint Probabilistic Data Association (JPDA) and Multiple Hypothesis Theorem (MHT) [33] [34]. The association method is selected the density of the data and complexity of the environment.

In the following subsections, the details of these steps will be given. In the first subsection, the basic concepts of Kalman filter and implemented Unscented Kalman Filter are introduced to understand the state estimation. Then, usage of multiple models in the state estimation is explained. In the final subsection, data association methods are represented. The end of this section, overall understanding of the tracking part will be finished.

3.3.1 State estimation

3.3.1.1 Kalman filter

Kalman filter is invented as a filtering and estimating technique for linear systems [6]. The filter starts with the initial belief of the states which must be Gaussian distributed with mean and covariance. The mean value of the belief is called the states and dynamic model is used for the evolution of the states. Then, states are combined with the measurements with the gain computed from the uncertainties. Basically, Kalman filter calculates the best estimation for the states by using sensor measurements and model predictions. Two phases are involved in per each cycle of the filter algorithm; state prediction and measurement update/correction. The states which are corrected from previous timestamp are used in the state prediction part. Then, the predicted output states and the sensor measurements are used on the correction part. Therefore, these steps are build a cycle. The state prediction and correction cycle is similar to Figure 3.8. In the prediction, previous cycle's output states are multiplied by system

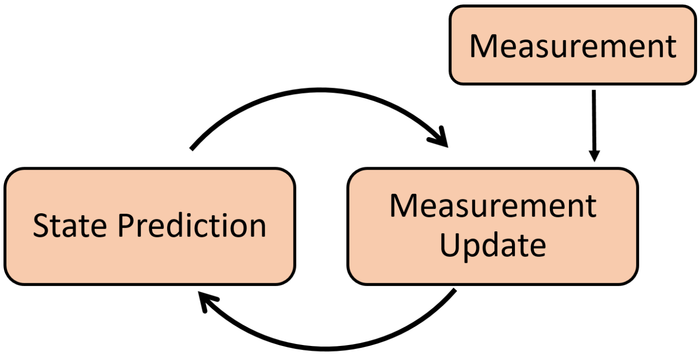

Figure 3.8: Kalman filter cycle.

matrix which depends on the system model. The system model are designed or selected from a designer according to the motion of the object. However, system model cannot reflect the motion exactly and there should be uncertainty on the transition. This uncertainty is added as Gaussian noises on the state prediction and as model error 'Covariance Matrix' on the propagation of the state covariance prediction. Thus, the error covariance of the states are also evaluated each cycle to understand the output states' uncertainty. The prediction step equations are given in (3.2).

$$\begin{aligned} X_t &= A_t X_{t-1} + B_t u_t, \\ P_t &= A_t P_{t-1} A_t^T + Q_t, \\ Z_t &= H_t X_t. \end{aligned} \tag{3.2}$$

X_t and X_{t-1} represent the state vector at current time t and previous time t-1. The state covariance's are P_t and P_{t-1} , respectively . A_t and B_t are the state transition matrices. The uncertainty of the state transition model is added to state covariance as Q_t in the prediction state. H_t which is the measurement model is used for measurement prediction according to the predicted state. The optimal Kalman gain (W_t) is computed as in (3.3).

$$W_t = P_t H_t^T (H_t P_t H_t^T + R_t) \tag{3.3}$$

where R_t represents the measurement error covariance which can be obtained from the sensor error characteristics.

The predicted state and covariance are updated with Kalman gain and the measurements as in (3.4).

$$
\begin{aligned}
X_t &= X_t + W_t(Z - Z_t),\\
P_t &= (I - W_t H_t)P_t.
\end{aligned}
\tag{3.4}
$$

The Kalman filter is an optimal filter when the Gaussian distribution will never change. All models must be linear in order to keep this distribution. However, the linear models are not sufficient to express all motions in practice. So, non-linear models must be implemented for transition of the states. For nonlinear system models, different applications of Kalman filter algorithms have been developed to handle this situation, such as Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF). The comparison of Kalman filters are represented in the Figure 3.9. In this figure the outputs of the each Kalman methods are illustrated with the same non-linear transition model. In the EKF algorithm, nonlinear models are linearized via Taylor Expansion [6]

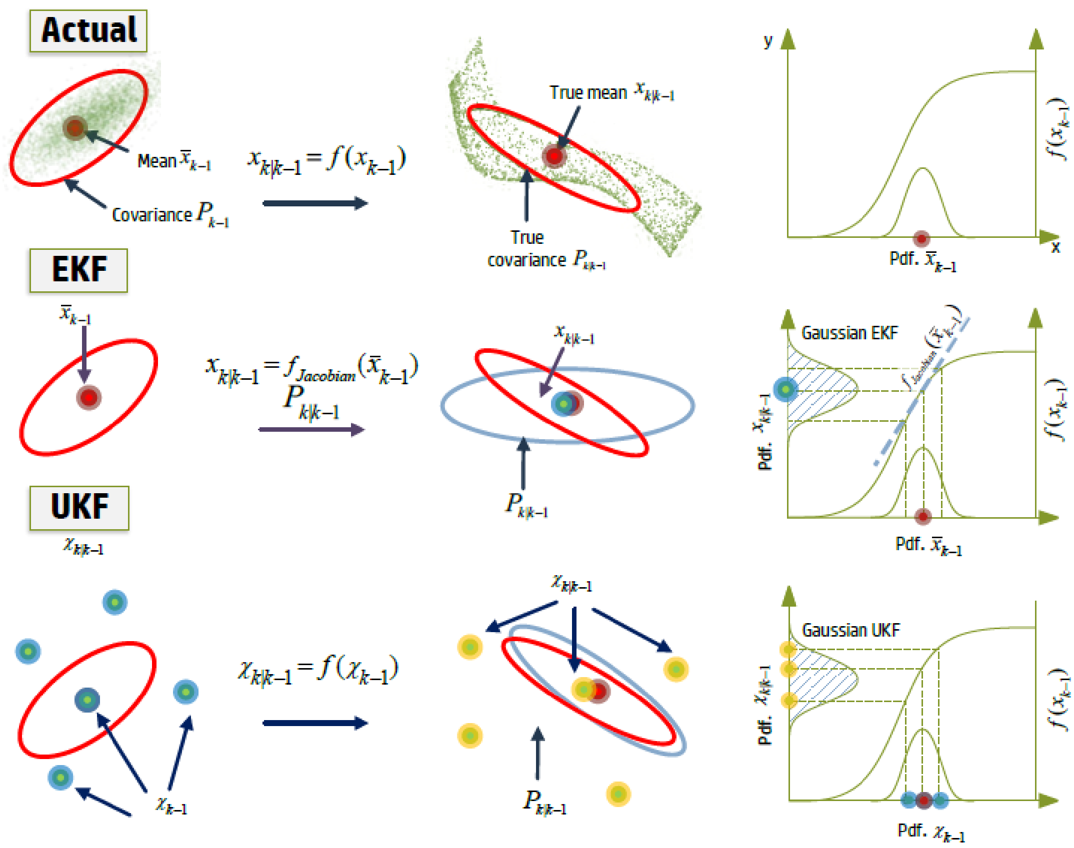

Figure 3.9: Comparison of LKF, EKF, UKF [6].

in order to use these models directly in the Kalman Filter equations. Before the prediction step in each cycle, Taylor expansion provides linear approximation of the non-linear function. However, this linearization may not be preferred due to the some drawbacks. The first drawback is that the cost of the calculation increase the total run time. The other main disadvantage is that the highly nonlinear models cannot be linearized properly.

The second extension of the Kalman Filter, UKF, works on non-linear models without using linearization methods. The key idea of UKF is based on approximation with sigma points. The sigma point are selected around the mean and are propagated through non-linear function. Then, best fit Gaussian is found from transformed sigma points. The UKF gives the best solution when highly non-linear models are used. It is also the most used type of the Kalman filter on multi-object tracking [3, 5, 12, 29]. Because of this, UKF algorithm steps will be detailed in the below. The UKF algorithm starts with the selecting sigma points around the mean and the sigma points have the equal weights in the beginning.

$$\begin{aligned}
&X^0_{t-1|t-1} = X_{t-1|t-1}, \\
&X^i_{t-1|t-1} = X_{t-1|t-1} + (\sqrt{(L+\lambda)P_{t-1|t-1}}), \qquad i = 1, ..., L \\
&X^i_{t-1|t-1} = X_{t-1|t-1} - (\sqrt{(L+\lambda)P_{t-1|t-1}}), \qquad i = L+1, ..., 2L.
\end{aligned} \tag{3.5}$$

where L and λ are design parameters. L value decides the number of sigma points and λ decides how much spread these points. The weights of $2L+1$ sigma points are defined both states and covariance's.

$$\begin{aligned}
W^0_s &= \frac{\lambda}{L+\lambda}, \\
W^0_c &= \frac{\lambda}{L+\lambda} + (1 - \alpha^2 + \beta), \\
W^i_s = W^i_c &= \frac{\lambda}{2(L+\lambda)}.
\end{aligned} \tag{3.6}$$

where α and β are scaling parameters. After sigma points determined, they are propagated directly with non-linear transition function as equation given in (3.7).

$$X^i_{t|t-1} = f(X^i_{t-1|t-1}) \qquad i = 0, ..., 2L \tag{3.7}$$

Propagated sigma points are combined again to produce the one predicted state matrix and covariance matrix.

$$\begin{aligned}
X_{t|t-1} &= \sum_{i=0}^{2L} W_s^i X_{t|t-1}^i, \\
P_{t|t-1} &= \sum_{i=0}^{2L} W_c^i [X_{t|t-1}^i - X_{t|t-1}][X_{t|t-1}^i - X_{t|t-1}]^T + Q_{t-1|t-1}.
\end{aligned} \tag{3.8}$$

After prediction step, measurement is predicted for the update state with the same way on state prediction part. The set of sigma points are obtained from predicted state (**??**) and then, predicted measurement is calculated by observation function (3.10).

$$\begin{aligned}
X_{t|t-1}^0 &= X_{t|t-1}, \\
X_{t|t-1}^i &= X_{t|t-1} + (\sqrt{(L+\lambda)P_{t|t-1}}), \qquad i = 1, ..., L \\
X_{t|t-1}^i &= X_{t|t-1} - (\sqrt{(L+\lambda)P_{t|t-1}}), \qquad i = L+1, ..., 2L.
\end{aligned} \tag{3.9}$$

$$Z_{t|t-1}^i = h(X_{t|t-1}^i) \qquad i = 0, ..., 2L \tag{3.10}$$

Predicted measurements are combined to form one predicted measurement (3.11).

$$Z_{t|t-1} = \sum_{i=0}^{2L} W_s^i Zt|t-1^i, \tag{3.11}$$

The variance of the residual which is named as residual covariance and the cross covariance between prediction and the measurement are computed to get UKF gain as given (3.12).

$$\begin{aligned}
P_{z,z} &= \sum_{i=0}^{2L} W_c^i [Z_{t|t-1}^i - Z_{t|t-1}][Z_{t|t-1}^i - Z_{t|t-1}]^T, \\
P_{x,z} &= \sum_{i=0}^{2L} W_c^i [X_{t|t-1}^i - X_{t|t-1}][Z_{t|t-1}^i - Z_{t|t-1}]^T, \\
K_t &= P_{z,z} P_{x,z}^{-1}.
\end{aligned} \tag{3.12}$$

The state and covariance matrices are updated with UKF gain as equation (3.13).

$$\begin{aligned}
X_t &= X_{t|t-1} + K_t(Z - Z_{t|t-1}), \\
P_t &= P_{t|t-1} - K_t X_t K_t^T.
\end{aligned} \tag{3.13}$$

3.3.1.2 Interacting multiple model

The other main part of the state estimation is determine the motion model of the object for prediction step. As mentioned above, using multiple models provides improved estimations. Interacting Multiple Model (IMM) provides a switching mechanism between different motion models according to the sensor measurements [32]. This subsection will give information about the IMM process.

In the beginning of the algorithm, the object motion models are defined with equal probability for all objects. The probabilities change over time according to the received measurement. This probability is called as a mode probability (μ) which means the probability of being right motion model of tracked object. Mode probability is updated each cycle with the update state. The steps of IMM process are given in the Figure 3.10. As seen in the figure, IMM algorithm starts with computation of initial states of each

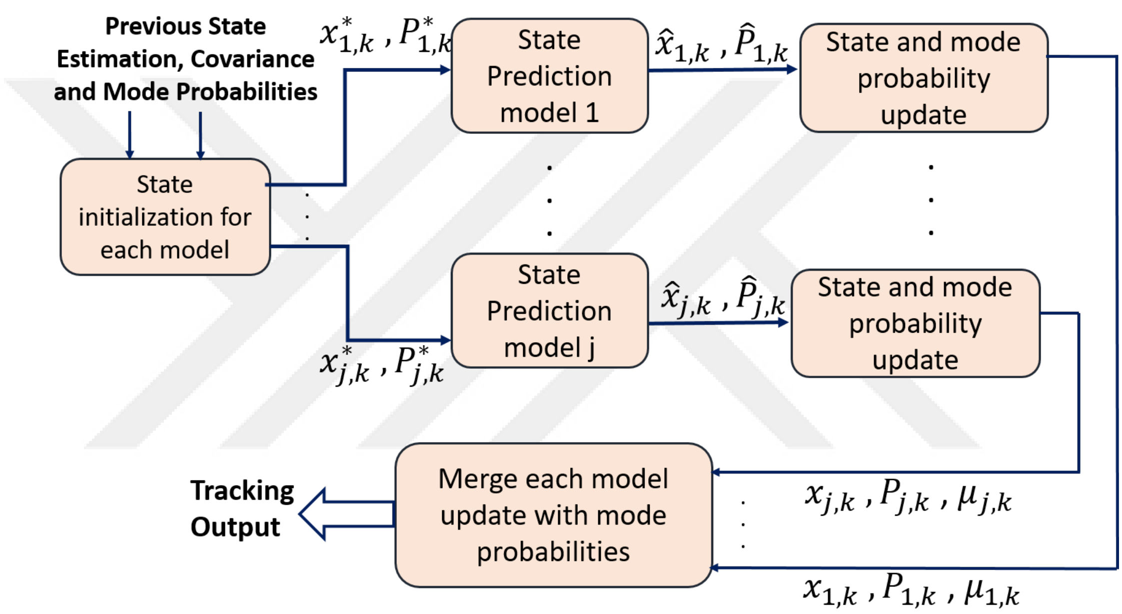

Figure 3.10: IMM workflow.

model. In this step which is called as mixing step [32], the initial state and covariance of track for each model j are computed as in 3.14. The total number of the selected model is shown as r.

$$\begin{aligned} X_{j,t-1|t-1} &= \sum_{i=1}^{r} \mu_{(j|i),t-1} X_{i,t-1}, \\ P_{j,t-1|t-1} &= \sum_{j=1}^{r} [P_{i,t-1|t-1} + (X_{i,t-1} - X_{j,t-1})(X_{i,t-1} - X_{j,t-1})^T] \mu_{(j|i),t-1} \end{aligned} \tag{3.14}$$

where $\mu_{(j|i),t-1}$ represents the conditional mode probability which implies the probability of transition model ith model to jth model and is computed as:

$$\begin{aligned} \mu_{(j|i),t-1} &= \pi_{ji} \mu_{j,t-1} / \mu_{i,(t|t-1)}, \\ \mu_{i,(t|t-1)} &= \sum_{j=1}^{r} \pi_{ji} \mu_{i,t-1} \end{aligned} \tag{3.15}$$

At the end of the mixing step, the initial states and covariance of each model are predicted with associated transition model. The prediction and update algorithms are implemented according to the selected Kalman filter method. After update step, model probabilities are updated.

$$\mu_{i,(t)} = (\mu_{i,t|t-1} L_{i,t}) / (\sum_{j=1}^{r} \mu_{j,t|t-1} L_{j,t}) \tag{3.16}$$

$L_{j,t}$ is the model likelihood with calculated from measurement and measurement residual covariance. In the final step of the IMM algorithm, there are updated states and covariance for each model and they are needed to be merged by mode probabilities to form the state estimation output as is done in mixing step.

3.3.2 Data association

Multi-object tracking not only predicts states but also deals with choosing the right measurement for state updates. This measurement selection process is called the data association is required either tracking with one sensor or multiple sensors. The reason why this process is required is that the number of objects is not known for many cases in the environment, and the source of sensor measurements is affected by noise, weak signals, and nearby targets. The cluttered environment causes multiple measurements to be associated for the tracked object. Also, different tracked objects can associated with the same measurements. The Figure 3.11 can be an example for the association problem. Two targets in the figure are going to close each other during tracking so measurements are close. Therefore, it is important to choose the right measurement

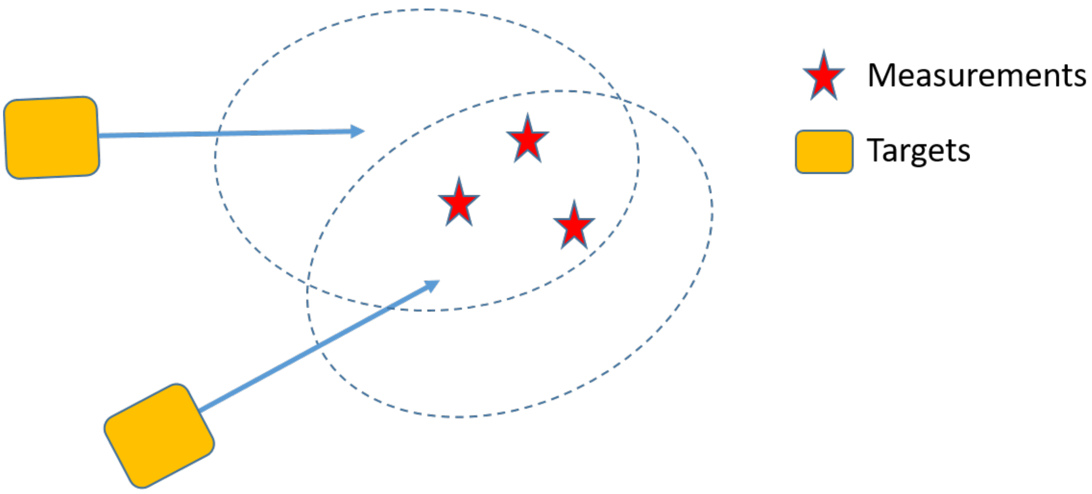

Figure 3.11: Data association problem example.

among them to track the target correctly because if the wrong measurement is used for the update, it will cause inconsistent results or even losing the target.

There are several approaches in the literature to solve the data association uncertainty and the common methods are Nearest Neighborhood(NN), Global Nearest Neighborhood(GNN), Probabilistic Data Association Filter(PDAF), Joint Probabilistic Data Association Filter (JPDAF) and Multiple Hypothesis Theorem (MHT) [35–37].

The NN method associates the measurement closest to the tracked object, and the GNN-based approach is an expanded version of the NN approach for multiple objects. This approach is one of the most common solutions for multiple target tracking problems because it is much easier to implement compared to many other methods. In this method, briefly, objects are associated with the closest measurement in a region that is determined around the object, the distance between measurements in the validation region and the tracks are calculated. Measurements are selected to minimize the sum of the distance for all tracked objects [37]. In the GNN, the same measurement is not shared with multiple objects, and each object is associated with only one measurement. However, these algorithms will fail if the targets are too close together [37]. Since the measurements of both targets are very close to each other, incorrect assignment can be made, which causes invalid state estimation.

One way to solve the data association problem is to calculate the probabilities that correspond to the relationship between the candidate measurements and the target. Probabilistic methods are based on using all measurements in the validation region instead of the best matched measurement. This logic is applied in PDAF algorithm [38] which details of the algorithm are explained through a single target tracking. This data association approach is applied for multiple objects in this thesis.

PDAF algorithm runs at the end of the prediction part. The observed measurements are evaluated with the target's predicted states. The calculation of the probability of the association between target and measurements is performed by using only the observations inside the validation region. Since ignoring the measurements outside of the validation region of a track reduces the complexity and computational effort. First of all, the Mahalanobis distance [39] between the measurements and the predicted measurement is calculated by using the residual covariance (S_t).

$$d = [Z - z_{t|t-1}]^T S_t^{-1} [Z - z_{t|t-1}] \tag{3.17}$$

The validation region is determined comparing gate threshold corresponding to the gate probability and Mahalanobis distances [40]. The set of validated measurements (m) are determined after this comparison. The association probability β_i between each validated measurements and the target is calculated. The probability β_0 which represents the none of the measurement is related to the target is also calculated (3.18).

$$\begin{aligned} \beta_i &= \frac{L_i}{1 - P_D P_G + \sum_{j=1}^{m} L_j} \qquad i = 1, ..., m \\ \beta_0 &= \frac{1 - P_D P_G}{1 - P_D P_G + \sum_{j=1}^{m} L_j} \qquad i = 0 \end{aligned} \tag{3.18}$$

where P_D is the target detection probability. It is assumed that this probability is known constant and independent over time. L_i represents the likelihood ratio of the measurement originating from the target. The computation of the likelihood based on the normal distribution around the predicted measurement (3.19).

$$L_i = N[z_i, z_{t|t-1}, S_t] P_D / \lambda \tag{3.19}$$

where λ is the density of Poisson process. The association probabilities are used to combine validated measurements. At the end of the algorithm one combined innovation is computed for updating the states of each track.

$$V = \sum_{i=1}^{m} \beta_i v_i \tag{3.20}$$

3.3.3 Multi-object tracking architectures in literature

Different combinations of the tracking methods can be found in literature. The [41] uses IMM-UKF-PDA and [42] uses IMM-UKF-MHT for multi object tracking. The JPDA-IMM-UKF combination is applied in [5] and [12]. The most used tracking algorithms for multi-object consist of IMM-UKF based state estimation and probabilistic data association.

4. FUSION APPROACHES FOR MULTIPLE 3D LIDAR SENSORS

In this thesis work, our problem is to decide how to fuse Multiple LIDAR sensors to get more precise detecting and tracking objects. Data fusion is a large research area which has various applications for multiple sensors. In order to decide the data fusion approach, the type of data used from the sensors must be decided. We can express the sensor data on three types as given in the Figure 4.1. The sensor outputs obtained

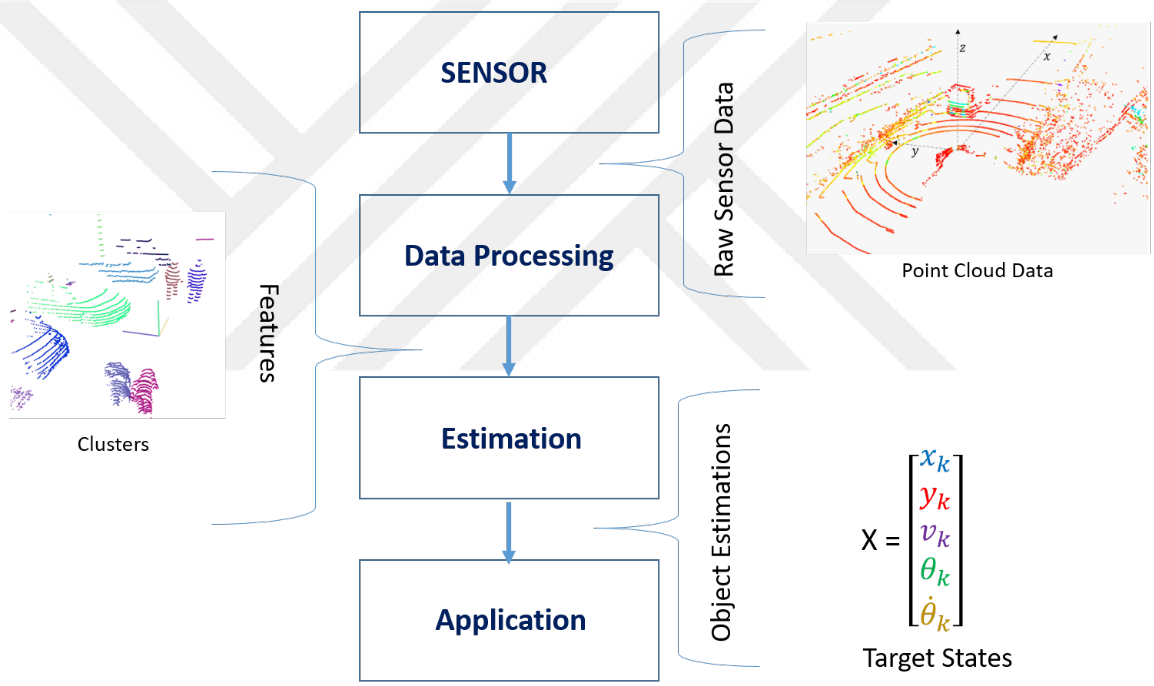

Figure 4.1: Sensor data abstraction levels.

without any operation are called raw sensor data. If any processing algorithms such as signal processing, clustering etc. are applied to the sensor raw data, this output is called detected features or clusters. When the Bayesian algorithm is applied to the cluster outputs, they are called track data. These tracks mean predictions of objects. The fusion algorithm varies according to which level of this sensor data will be combined. In general, two basic approaches to sensor fusion have been studied; low level data fusion and high level data fusion.

Low-level sensor data includes unprocessed raw data and features. Fusing low-level information is performed as combining raw data and then processing this combined raw data. After fusion algorithm, Bayesian estimator is used to get tracking outputs. This approach minimize the information loss but increase the data so increase the complexity. Object estimations are obtained by applying Bayes estimator to the processed sensor data and then fusion algorithms are applied to these estimates. This fusion approach is called a high level data fusion or track-to-track fusion. High level fusion is the most popular way of multisensory data fusion for autonomous driving. The most general form of the track-to-track fusion is combining tracked objects without using any correlation information between the tracks.

In this study, these two fundamental fusion approaches are implemented to the multiple 3D LIDAR sensors. As a low-level fusion approach, the point cloud data from each sensor is combined and ODT is applied on the merged cloud. The low-level fusion of multiple LIDAR sensors is already studied well-known process in the literature. On the other hand, this thesis is proposing and implementing "high level track-to-track fusion" method on multiple 3D LIDAR sensors. The proposed method evaluated is to find objects separately from each sensor data and to implement high level track to track fusion method. These two approaches are investigated for the appropriate one by evaluating the results. In the following section the implemented ODT algorithm will be explained in detail. Next, applied fusion approaches will be given.

4.1 Selected Multi-Object Detection and Tracking Architecture

The detection and the tracking algorithms are implemented in the same way for both sensor fusion approaches which used in the thesis. The implemented algorithms for the detection process and the tracking process are represented in Figure 4.2. The object

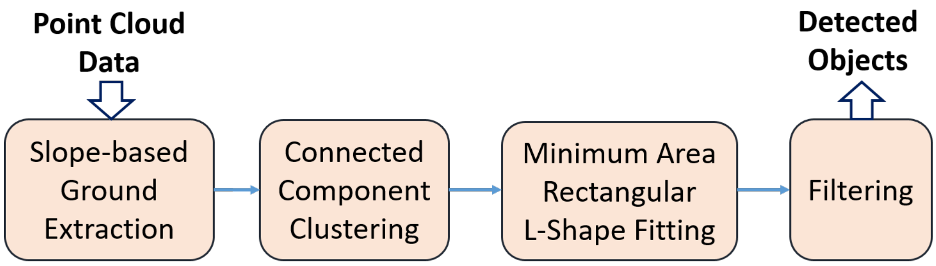

Figure 4.2: Detection workflow.

detection workflow starts with ground extraction. Ground points are removed firstly with slope-based ground extraction algorithm. This algorithm uses height values of the points. The known sensor height decides the minimum and maximum threshold to decide which points are checked. Then, each point's height value is compared with the adjacent points. If the height difference or slope between two points is lower than the defined threshold, these points are evaluated as the ground points. A sudden increase of height shows the non-ground points [18]. The Figure 4.3 displays the point cloud data by combining the bird's-eye view of three different times for the same scenario after the ground points have been removed. This figure also helps to understand the points falling on vehicles when their target vehicles move and their location changes. Non

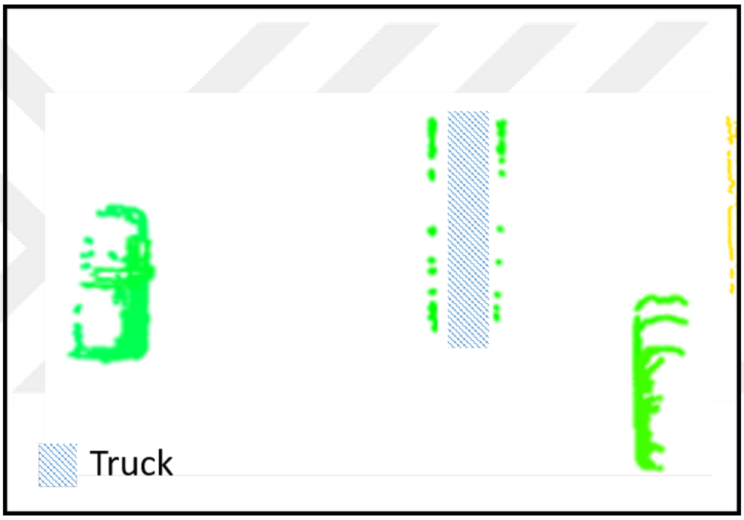

Figure 4.3: Bird's-eye view of point cloud data after ground is extracted. The scenario includes two vehicles that move straight on either side of the truck.

ground point cloud data is projected to occupancy grid. Empty grids are labelled as (0) and occupied grids are (-1). Then, the occupied grids are iteratively checked with all adjacent grids, and if they are close to each other, they are labeled as a same cluster ID as seen in the Figure 4.4. This process is repeated until all occupied grids are labelled with a cluster ID. Next, Minimum Area Rectangle (MAR) and L-shape fitting methods are applied to each cluster to obtain the bounding boxes. MAR algorithm firstly finds a rectangular shape from clusters in 2D grid cells. But this rectangular does not always fit the objects, L-Shape fitting corrects the orientation of the rectangular. After getting 2D rectangular shapes, the height values of the points construct the 3D boxes.

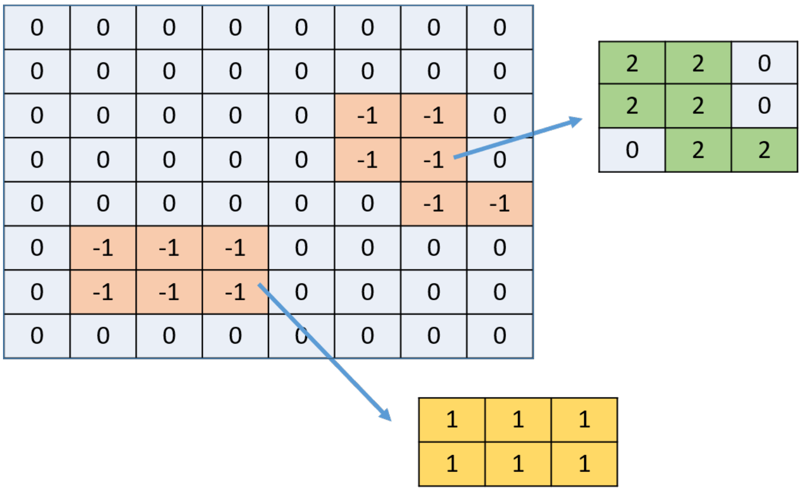

Figure 4.4: Clustering with occupancy grid.

The last step of the detection workflow is the filtering. The bounding boxes are filtered to get rid of unrelated objects such as buildings trees etc. The filter algorithm checks the following properties of the bounding boxes;

- number of points into the box
- the volume of the box
- the width-height ratio of the box

After filtering, the obtained objects from the point cloud data is illustrated in the Figure 4.5. The output of the detection process provides the corner points of the bounding boxes. The center point is calculated from the corner points for tracking process. Detection algorithm finds the objects in each scan and does not include the history information. More than two states can be found with the help of the previous measurements and the history of the object. The tracking algorithms provide a detailed object information. In the tracking algorithm, objects are represented by state vector. The state of the objects at time stamp k includes relative positions x_k y_k, velocity v_k, yaw ψ_k and yaw rate $\dot{\psi}_k$ information. Thus the state vector is defined as illustrated in (4.1). In this thesis, ISO coordinate system was used.

$$X_k = [x_k \quad y_k \quad v_k \quad \psi_k \quad \dot{\psi}_k]' \tag{4.1}$$

The flow chart of the tracker algorithm is given in Figure 4.6. Multi-object tracking includes Interactive Motion Model (IMM) to estimate the dynamic behavior

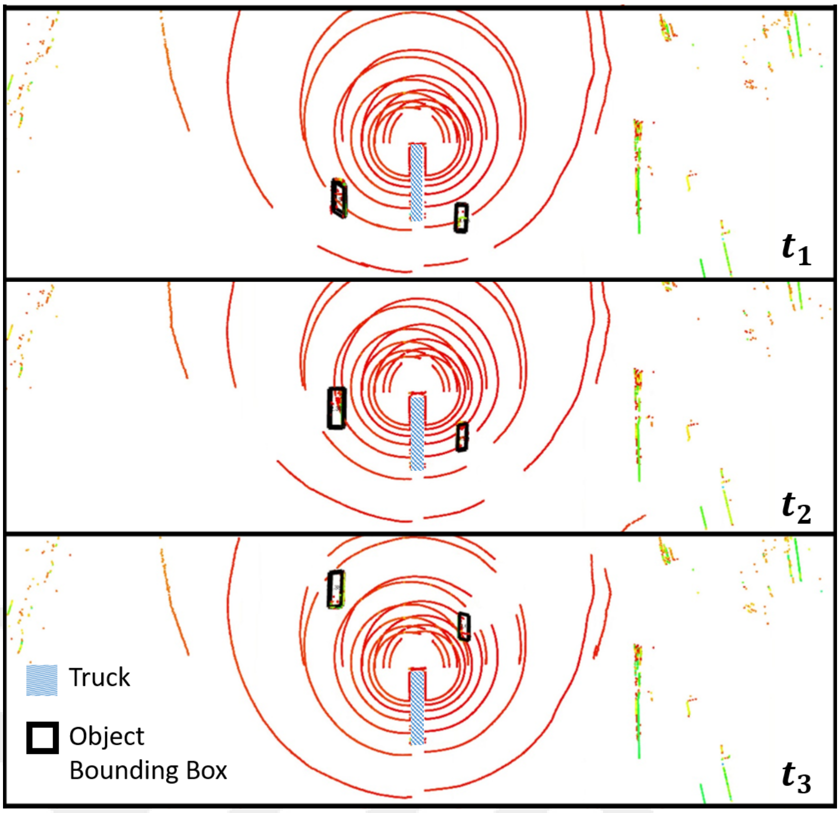

Figure 4.5: Bird's-eye view image of point cloud and bounding boxes of objects for the three different times of the same scenario (t is for time representation and $t_1 < t_2 < t_3$).

of the object, Unscented Kalman Filter (UKF) as a state estimator to process nonlinear motion models, and Probabilistic Data Association (PDA) filter to obtain the appropriate measurement for update. In the IMM estimation three motion models are used; Constant Velocity Model (CVM), Constant Turn Rate Velocity Model (CTRVM) and the Random Motion Model (RMM). In CV model, the velocity of the vehicle is assumed constant and the turn rate is assumed as zero. So, the motion model is obtained from the equations in (4.2).

$$\begin{aligned}
x_t &= x_{t-1} + \Delta t v_t sin(\psi_t), \\
y_t &= y_{t-1} + \Delta t v_t cos(\psi_t), \\
v_t &= v_{t-1}, \\
\psi_t &= \psi_{t-1}, \\
\dot{\psi}_t &= 0.
\end{aligned} \tag{4.2}$$

This model is sufficient when vehicle does not turn but if a target turns, CVM is not enough for tracking process. So, the CTRV model is selected to express turning cases which assumes target turns with constant turn rate and velocity. The transition

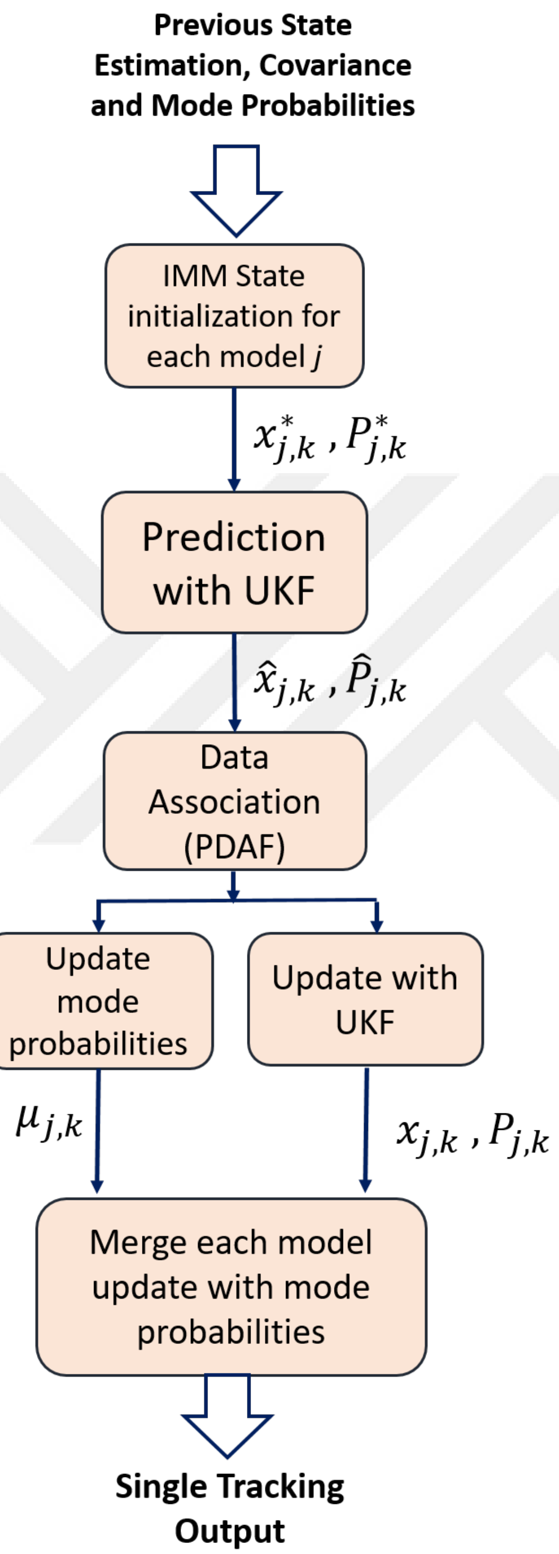

Figure 4.6: Tracking workflow.

equiations are given in (4.2).

$$\begin{aligned} x_t &= x_{t-1} + v_t/\dot{\psi}_t sin(\psi_t) - v_t/\dot{\psi}_t sin(\psi_{t-1}), \\ y_t &= y_{t-1} - v_t/\dot{\psi}_t cos(\psi_t) + v_t/\dot{\psi}_t sin(\psi_{t-1}), \\ v_t &= v_{t-1}, \\ \psi_t &= \psi_{t-1} + \Delta t \dot{\psi}_t, \\ \dot{\psi}_t &= \dot{\psi}_{t-1}. \end{aligned} \tag{4.3}$$

The last model which is named as random motion model, assumes there is no motion on the target object. This model is used for represents the static objects. Three estimations are always calculated separately with these motion models at each time stamp, as shown in (4.4).

$$\widehat{X}_{j,t} = f_j(X^*_{j,t-1}, u_t) + w_{j,t} \qquad j = 0,1,2..r \tag{4.4}$$

where j represents the model number, w is the noise. Each model has the mode probability $\mu_{j,t}$ which changes over time according to the measurement. In order to do prediction, each model's initial state is calculated using conditional mode probabilities $\mu_{(i|j),t-1}$ and the previous cycle updated state $X_{i,t-1}$ as given equations (4.5).

$$\begin{aligned} X^*_{j,t-1} &= \sum_{i=1}^{r} \mu_{(i|j),t-1} X_{i,t-1}, \\ P^*_{j,t-1} &= \sum_{i=1}^{r} \mu_{(i|j),t-1} [P_{i,t-1} + (X_{i,t-1} - X^*_{j,t-1})(X_{i,t-1} - X^*_{j,t-1})^T] \end{aligned} \tag{4.5}$$

UKF prediction steps which is mentioned in Chapter 3 are followed for each model prediction. After prediction part, measurements are evaluated for update part of the UKF. The PDA algorithm is used for data association. The workflow of the algorithm is given in Figure 3.11. This data association method computes the probability of each measurement in the validation region of the predicted state [38]. In this algorithm, the mahalanobis distance between the measurements and the predicted measurement is calculated using measurement covariance. The validation region is determined by a threshold, depending on the number of states. If the measurement distance is lower than the threshold, the probability of being right measurement is computed. By combining the innovations of these measurements with the probability of associating them, a single innovation value is created for update.

The predicted state and covariance of each model is updated with the combined innovation by UKF updated steps. Model probabilities are also updated by comparing

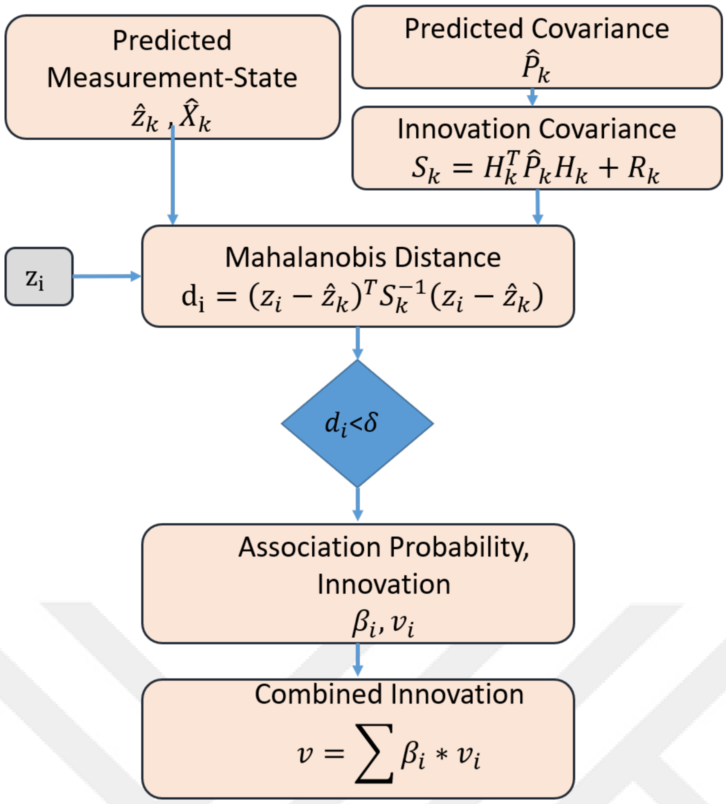

Figure 4.7: Probabilistic Data Association workflow.

with the associated measurement. In the final step of the cycle, updated states and covariances of each model are combined with the mode probabilities in order to obtain single verified information of the object.

4.2 Low-Level Fusion Approach of Multiple 3D LIDAR

This section describes the first solution approach which is Detection and Tracking After Merging Raw Data (DTAMR) method and its architecture is shown in Figure 4.8. DTAMR algorithm starts with the frame transformation of each point cloud data

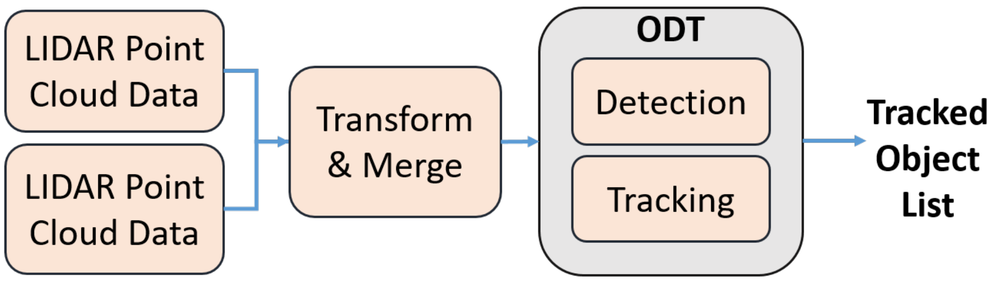

Figure 4.8: DTAMR architecture.

to the vehicle frame. Two point cloud data are merged with using Point Cloud Library (PCL) after transformation. End of the merging part, single dense point cloud data is obtained as seen in the Figure 4.9. This data is fed to the detection and tracking algorithm.

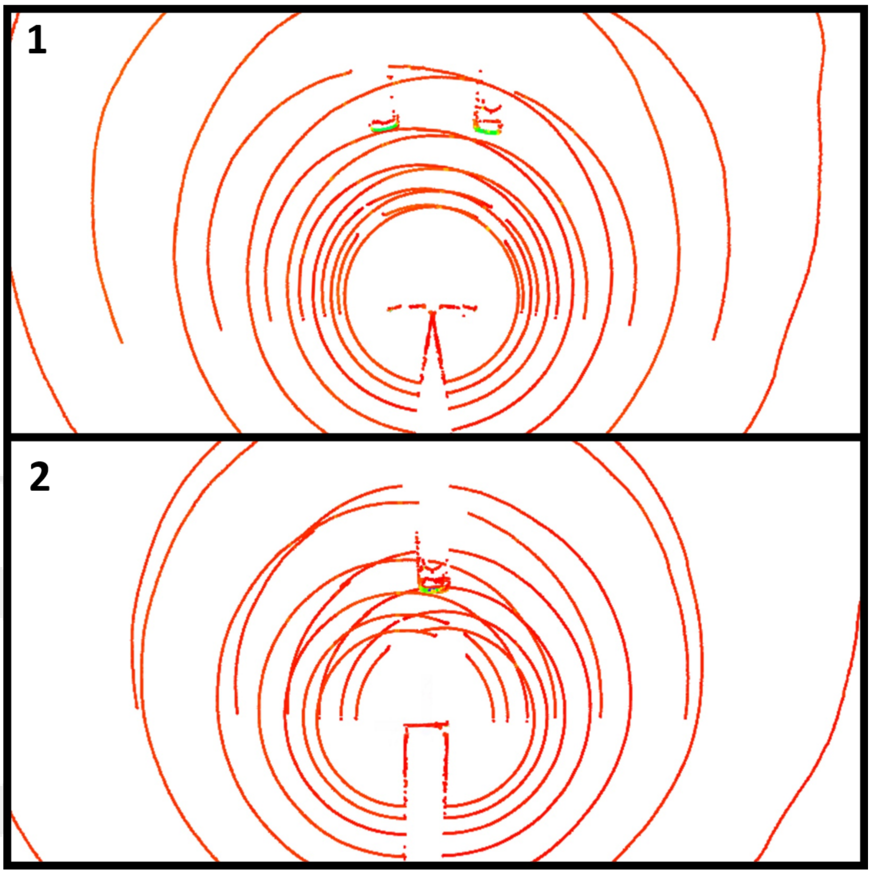

Figure 4.9: Birds-eye view of point cloud data before transformation (1) and after transformation and merge (2).

4.3 High-Level Fusion Approach of Multiple 3D LIDAR

The proposed approach is Track-to-Track Fusion of Multiple LIDAR (T2TFML) which is based on getting two tracking results from each LIDAR sensor and combining them with the track-to-track fusion algorithm. The proposed architecture is illustrated in Figure 4.10. The detection and the tracking algorithms are implemented for each LIDAR point cloud data. After ODT operations, two point cloud data gives tracked object states on LIDAR frame and the covariances of the object states. Then, tracked objects are transformed to the vehicle frame. There are two object detection and tracking outputs which are obtained from each LIDAR ODT algorithms. The example fusion inputs for one target vehicle is represented in the Figure 4.11. The center points of the objects are sent to the fusion. The fusion block which is shown in the last part of the workflow consists of three main parts, state estimation,data association

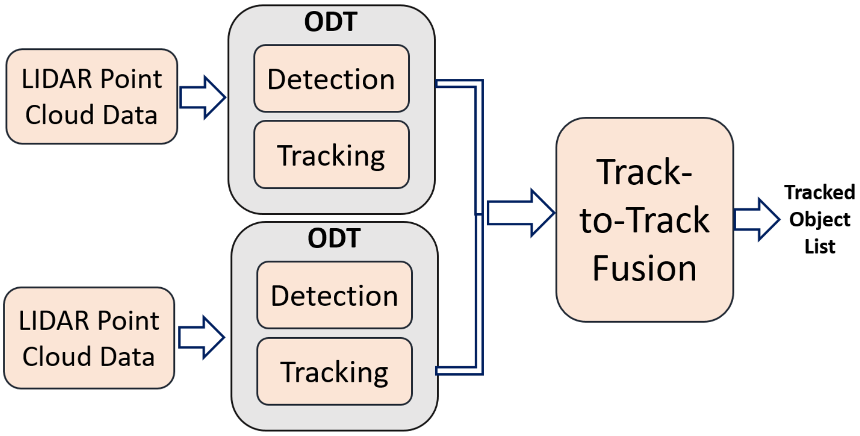

Figure 4.10: T2TFML architecture.

and track-to-track fusion method. At the beginning of the algorithm, a global tracked object list is created. This object list is predicted to the current time for each timestamp

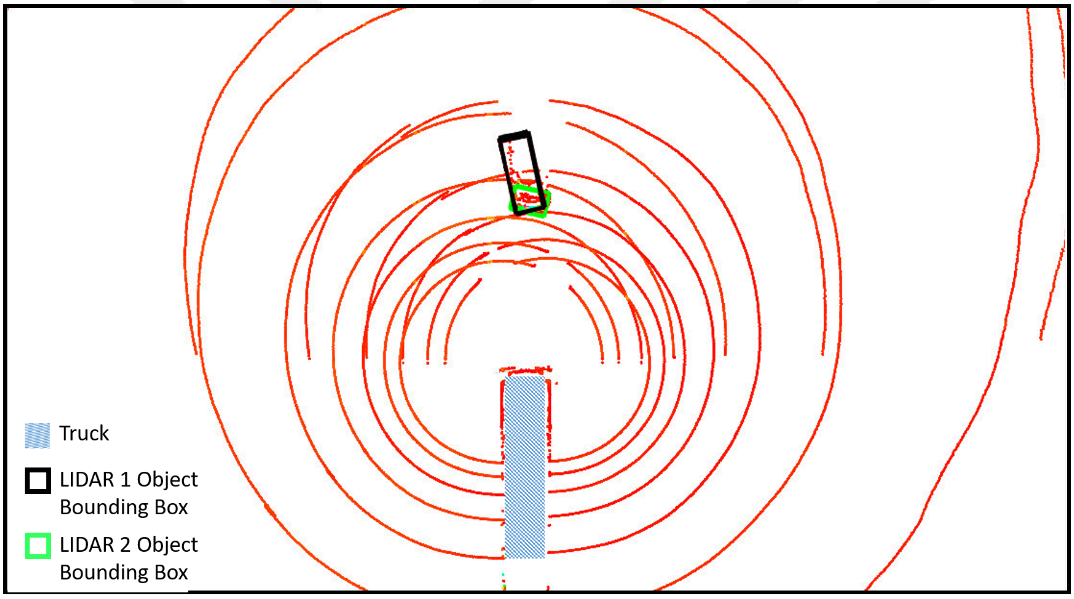

Figure 4.11: Birds-eye view point cloud data of two ODT output from each LIDAR sensor for target vehicle within intersection area (Black box represents the 1^{st} LIDAR sensor output and green box represents the 2^{nd} LIDAR sensor output).

with CTRVM based UKF estimation. Next, sensor measurements are compared with the predicted object states. The association method is used to decide which tracks are fused. Nearest Neighbor algorithm is used for data association and so the closest tracked object is associated with the global object.

Then, associated tracked object is fused with Covariance Intersection (CI) track-to-track fusion method as given equations in (4.6) [43].

$$
\begin{aligned}
P_f^{-1} &= w * P_1^{-1} + (1-w) * P_2^{-1} \\
X_f &= P_f(w * P_1^{-1} X_1 + (1-w) * P_2^{-1} X_2)
\end{aligned}
\tag{4.6}
$$

where X_1 and X_2 are sensor track states coming from two LIDARs, X_f represents fused state of the object, P_1 and P_2 are the state covariances of the object, P_f is the fused object covariance. Notice that, the covariances are combined first, then states are combined with this fused covariance. The sensor track states are multiplied with the weights which are seen as (w) and $(1-w)$ in the equations. The selection of the weight parameter is based on the lateral distance of the object to LIDARs. Different choices of weights can be used to optimise the fusion outputs.

5. EXPERIMENTAL SETUP

In this study, in order to test and compare the algorithms; measurements and the ground truth data are collected directly by mounting sensors on Ford F-Max heavy duty truck. Two Velodyne 16-channel (VLP-16) LIDARs are positioned on the front corners of the truck at the same height as the top of the passenger car (about 1.8 meters height). The locations of LIDAR sensors on truck are illustrated in the Figure 5.1. Each sensor

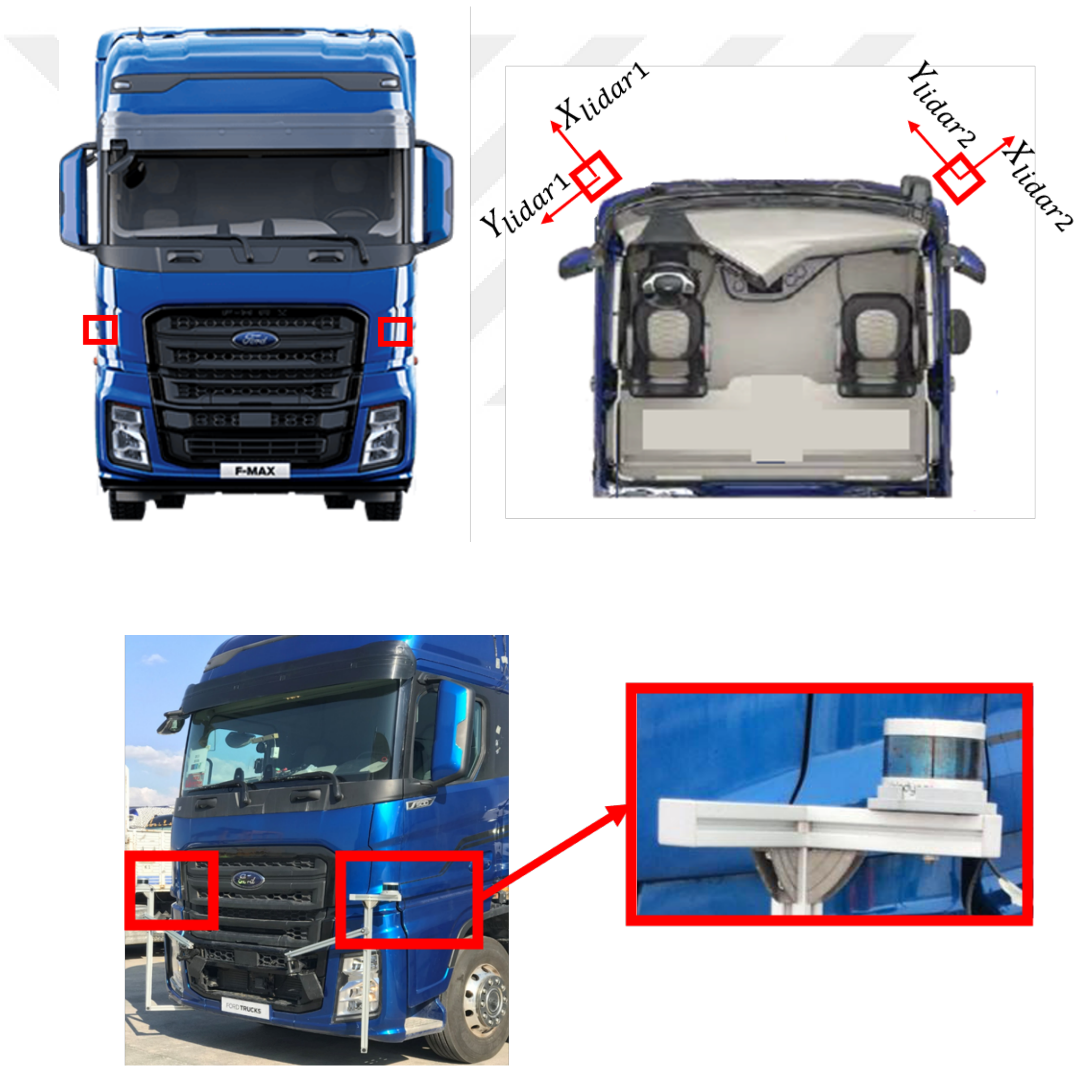

Figure 5.1: LIDAR locations and LIDAR sensor setup on truck.

covers around 270 degrees field of view. The front of the the vehicle, there is common area where both of the LIDARs' field of view intersects. The algorithms are tested and compared in this area. Figure 5.2 represents the LIDARs' field of views. LIDAR

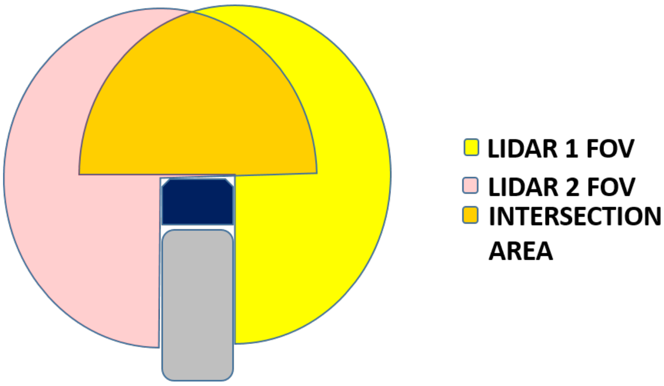

Figure 5.2: LIDAR sensor FOVs.

sensor generates 3D point cloud data with respect to their reference frame. The point cloud includes 300.000 points at each scan and works at 10 Hz frequency. Points store x, y, z position information and the intensity values. The implementation is written by using C++ programming language. The fusion applications run real-time on NVIDIA Jetson TX2 embedded platform. Operating System based on LINUX is used on this embedded platform. The Robot Operating System (ROS) is a middleware which rund on LINUX operating systems. ROS which is an open-source program provides a communication infrastructure. Also, there are open-source Velodyne LIDAR sensor drivers in ROS. These drivers captures the LIDAR data and publish in a topic as a ROS messages. The implemented algorithms subscribe this topic and obtain raw sensor data. The tracking outputs of the fusion algorithms are also published as a ROS messages. The Figure 5.3 shows the ROS communication architecture. Oxford

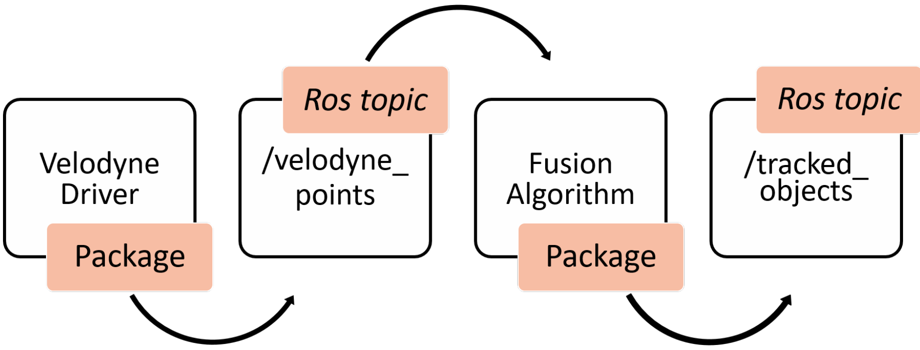

Figure 5.3: ROS communication architecture.

Technical Solutions'(OxTS) GNSS/INS systems are used in order to collect ground truth data. RT3000 GNSS/INS is placed on the truck and this sensor gives the position, orientation with high speed and with high accuracy. RT-Range is mounted to the target vehicle to get the accurate relative positions, relative velocities and orientations with respect to the RT3000. In Inönü-Eskisehir, Turkey, there is a private and fully-equipped testing ground that is within the factory and is closed to regular traffic. This proving ground is facilitated with base stations for GNSS which provides 1 cm accuracy for ground truth measurements.

6. REAL-WORLD TESTING & RESULTS

This section provides description of test cases and analysis of test results. Three different scenarios are tested on the proving ground to evaluate the applied sensor fusion methods. The scenarios are illustrated and described in Figure 6.1. In order

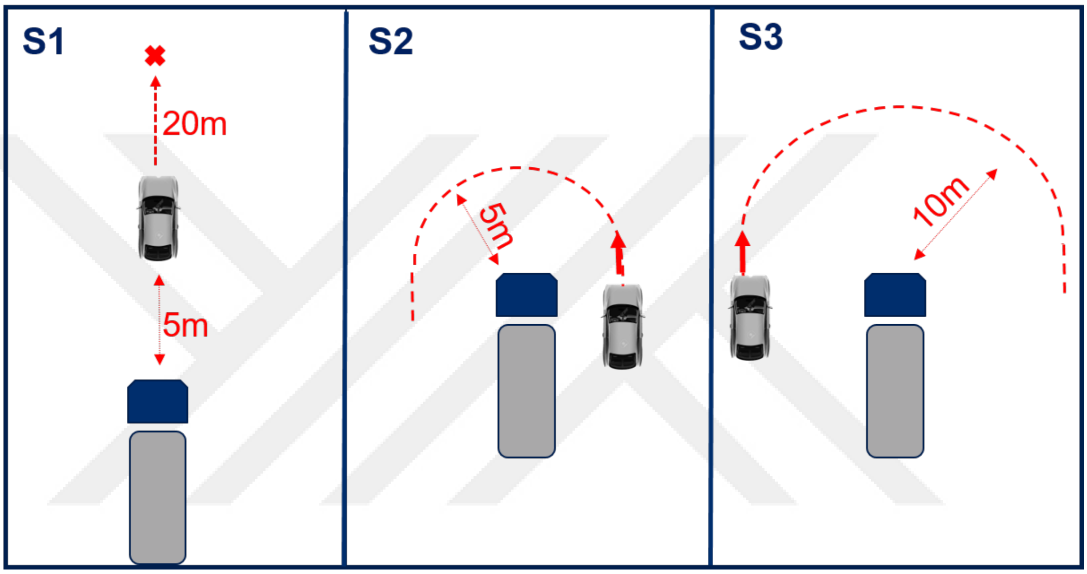

Figure 6.1: Scenario 1: Target vehicle moves from 5m to 25m on the front of the truck, Scenario 2: Target vehicle moves around the truck on circle with a radius of 5 meters, Scenario 3: Target vehicle moves around the truck on circle with a radius of 10 meters.

to get more accurate fusion results, adaptive weight (w) parameter which is given in equation (4.6) was tuned. During our tests, the value of weight parameter is selected between 0.3; 0.5 and 0.7.

The comparison of the estimated object states (relative longitudinal distance, lateral distance and velocity) for two fusion approaches with ground truth are shown for three scenarios. Figure 6.2.shows the state results for the Scenario 1. The detailed analysis of Figure 6.2 is given as follows. In the beginning of the test, it can be seen from Figure 6.2 that T2TFML method has delayed around 1 second because of additional spent time to run the ODT algorithm before the fusion starts.

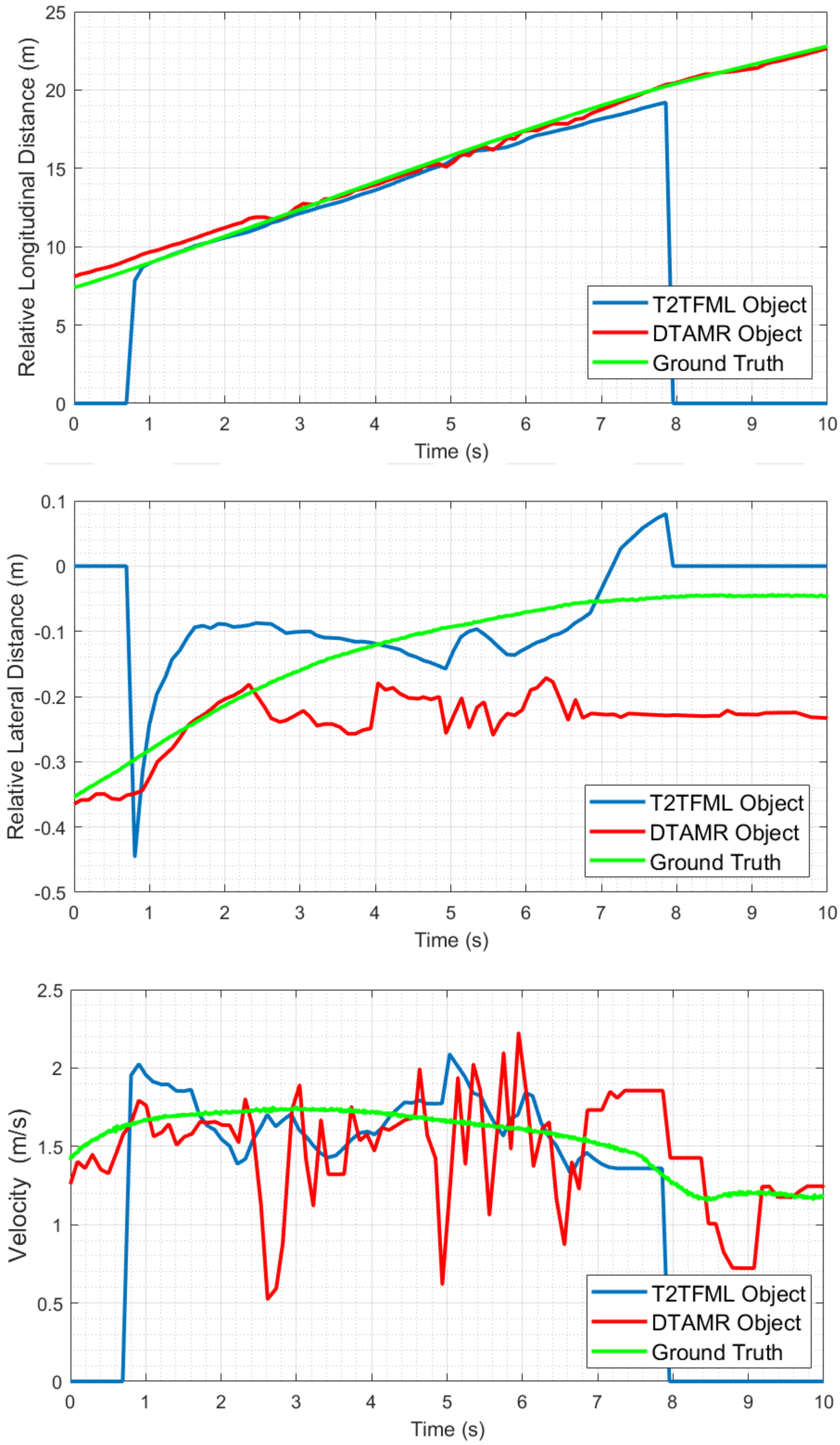

Figure 6.2: Relative states of the object with ground truth data for Scenario 1.

In order to achieve a fair comparison, DTAMR and T2TFML fusion results are evaluated in the intersection area of FOVs which can be seen in Figure 5.2. The track-to-track fusion provides more stable characteristic of velocity estimation than the low level fusion in that specific region.

At the end of the test (after 8 seconds), it can be seen in Figure 6.2 that T2TMLF approach has lost the object detection and tracking earlier than the DTAMR approach. DTAMR approach is using the merged point cloud data which are denser than the single LIDAR point cloud data used in T2TFML. In our test although the object moves far away, the number of points on the object does not decrease very quickly with merged point cloud data used in DTAMR. So, this allow DTAMR approach to continue and keeps tracking longer (until the end of the test). The Figure 6.3 gives the absolute errors of estimated object states for Scenario 1. The state estimates and absolute errors of the states for Scenario 2 is shown in Figure 6.4 and Figure 6.5, respectively. The results are compared after the target vehicle enters the intersection area completely. The target vehicle often makes lateral movements throughout the scenario. Even if the results give values close to each other, in the lateral direction estimates, track-to-track fusion approach gives results closer to ground truth values. In the velocity state, it is seen that the proposed method provides improvements on accuracy by increasing stability and reducing the error. On the other hand, the low-level fusion is better estimates than the track-to-track fusion on longitudinal position state.

Scenario 3 is also designed to test lateral state estimates, and this scenario involves tracking a target vehicle that moves farther than the target on Scenario 2. Thus, this test aims to show us whether the distance is effective in improving lateral estimates. When the results are evaluated, the same improvements in lateral position and velocity accuracy in Scenario 3 are seen with the track-to-track fusion approach. The state estimations and the absolute error graphs are given in Appendix A.1. The performance criteria for evaluating the two fusion approaches are execution time, mean square error, and standard deviation. The average execution time of the DTAMR is 27ms and the T2TFML is 36ms. Even if the execution time of T2TFML approach is 10ms longer, this does not cause any delay in the obtained results because of the working cycle of VLP-16 LIDAR which is 100ms. In order to evaluate all test scenarios, Table 1 contains a detailed comparison of the two approaches with the two performance

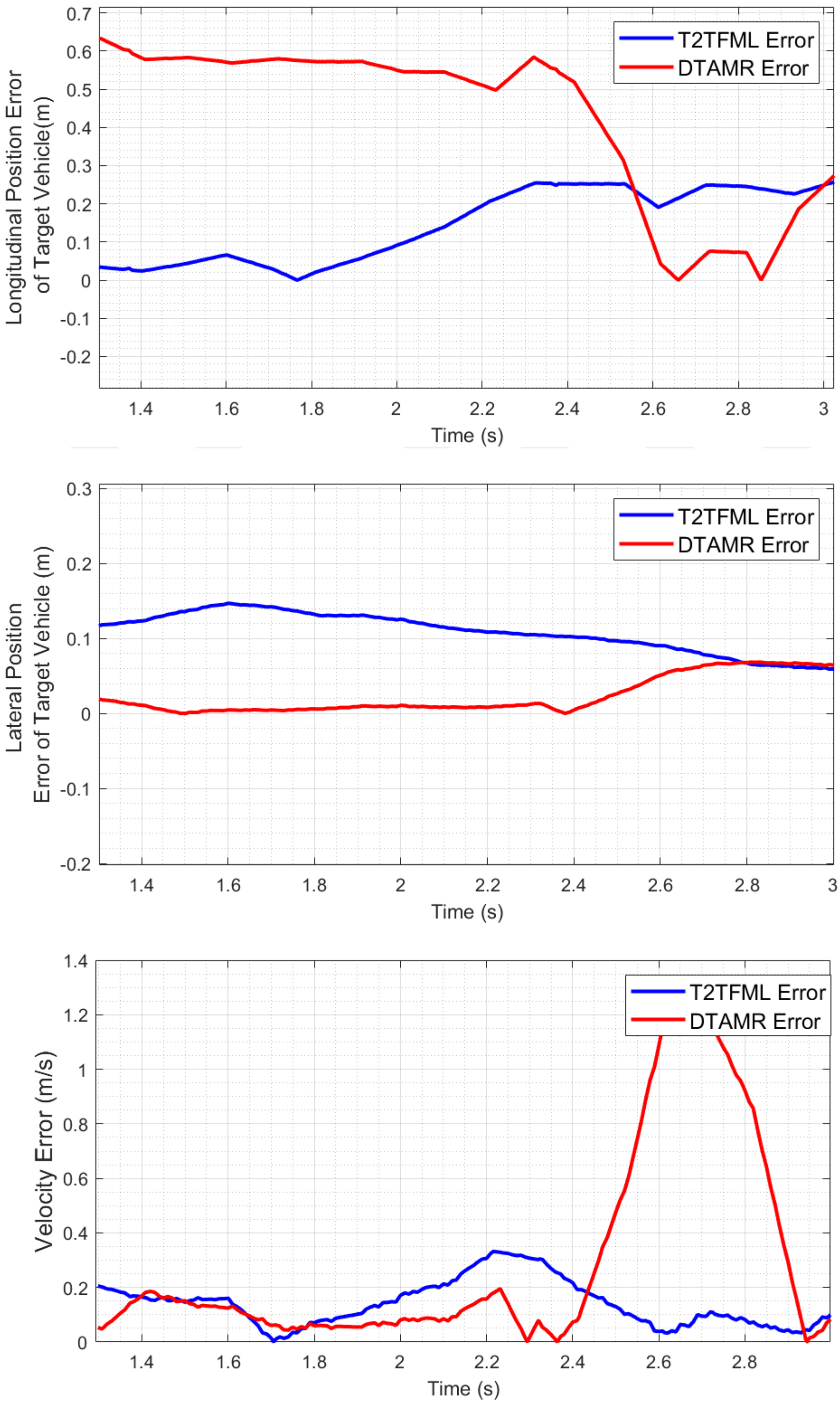

Figure 6.3: Absolute errors of estimated object states for Scenario 1.

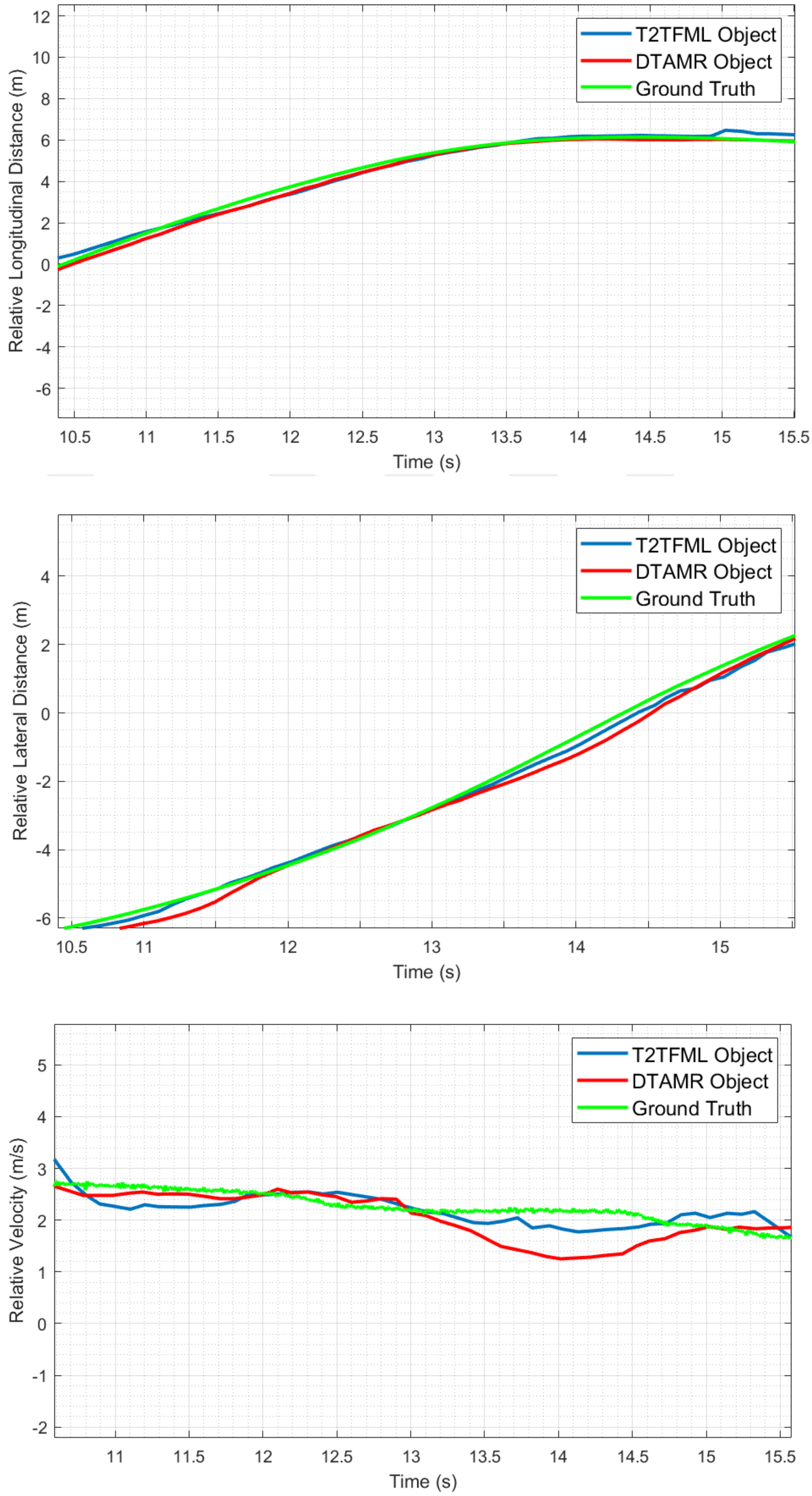

Figure 6.4: Relative states of the object with ground truth data for Scenario 2.

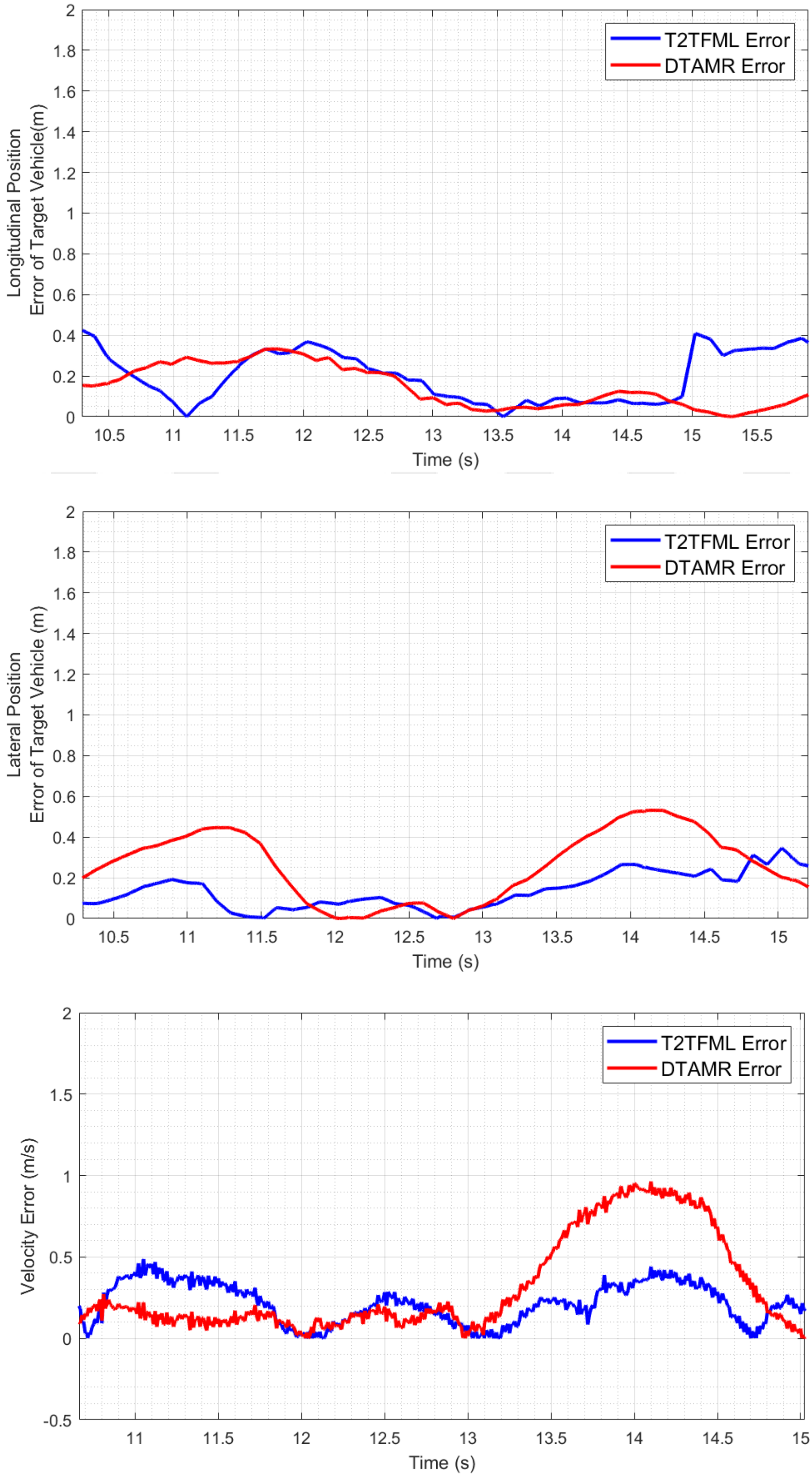

Figure 6.5: Absolute errors of estimated object states for Scenario 2.

metrics; Mean Square Error (MSE) and Standard Deviation (SD). MSE equation is given in (6.1) and provides information about the accuracy of the prediction. Standard deviation represents the distribution of results according to actual data (6.2).

$$MSE = \frac{1}{n}\sum_{t=1}^{n} e_t^2 \tag{6.1}$$

$$SD = \sqrt{\frac{1}{n-1}\sum_{t=1}^{n-1} e_t^2} \tag{6.2}$$

In Scenario 1, test is used for analysis of fusion tracking performance on longitudinal

Table 6.1: MSE and SD results of Scenarios

Scenario	Approach	Mean Square Error			Standard Deviation		
		Long. Distance (x)	Lat. Distance (y)	Vel. (v)	Long. Distance (x)	Lat. Distance (y)	Vel. (v)
Scenario 1	DTAMR	0,2230	**0.0011**	0.2185	0.4723	**0.0340**	0.4675
	T2TFML	**0.0302**	0.012	**0.0267**	**0.1738**	0.1112	**0.1634**
Scenario 2	DTAMR	**0.0288**	0.0645	0.1581	**0.1698**	0.2541	0.3977
	T2TFML	0.0291	**0.0286**	**0.1511**	0.1706	**0.1693**	**0.3887**
Scenario 3	DTAMR	**0.1107**	0.2165	0.0144	**0.3327**	0.4653	0.1201
	T2TFML	0.1245	**0.1668**	**0.0130**	0.3528	**0.4084**	**0.1140**

direction. The proposed algorithm follows the object in a shorter time than the low-level fusion approach. However, in specific region the accuracy of longitudinal position and the amplitude of the velocity of the object are increased by %86 and %88, respectively. Scenario 2 and 3 are used for analysis of tracking performance on lateral states. The results from the table show that the proposed approach improved accuracy on the tracked object lateral position estimation around %22 to %55 and vehicle velocity estimation around %4 to %10. The accuracy of the longitudinal position estimation is reduced within negligible range.

7. CONCLUSION AND DISCUSSION

Environment perception is a critical part of autonomous driving which requires to get a reliable and accurate object information from surround. LIDAR sensors are thought to be key enablers for autonomous cars through their significant advantages on wide field-of-view and high-resolution capabilities. Automotive companies' interest in LIDAR sensors is also thought to increase with slashed sensor prices over the years.

Our main aim in this research is to get more precise object detection and tracking (ODT) results using multiple LIDAR sensors in real time for autonomous vehicles. In the first part of the thesis the detection and tracking algorithms for LIDAR sensor are explained in detail. Then, the selected methods of ODT algorithm are explained and the results are represented. The output of the ODT algorithm provides bounding boxes of the objects and the tracked states of these boxes. The sensor fusion methods are studied in the rest of this work.

During this thesis, we have developed, applied and tested two different (low and high level) real-time sensor fusion methods on multiple 3D LIDAR sensors for object detection and tracking. The first contribution of this work is proposing and implementing "high level track-to-track fusion" method on multiple 3D LIDAR sensors. In the presented high level fusion, tracked objects from each LIDAR sensor are combined with Covariance Intersection fusion method. To the best of our knowledge, this is the first automotive application of track-to-track fusion method on multiple 3D LIDARs. Within the research we have also applied and tested well known and studied low level fusion of multiple 3D LIDARs.

In the low-level fusion approach, the raw point cloud data from each LIDAR sensor is directly fused and ODT algorithms are applied on this fused data. These two fusion approaches include the object detection and tracking algorithm which is explained in detail. Another contribution is the analysis and comparison of track-to-track fusion method performance with the well-studied low-level real-time fusion method. These two real-time fusion strategies are implemented in the experimental test truck which is instrumented with two 3D LIDAR sensors. The front of the vehicle is adjusted as an intersection area for testing fusion approaches. The performance of the fusion strategies are tested under three different driving scenarios which are designed for testing lateral and longitudinal states. Additionally, the ground truth data is collected with the help of the global navigation satellite system (GNSS) in high accuracy for performance evaluation. The test results are analyzed in terms of defined performance criteria and the benefits & weaknesses of the proposed approach are discussed in this work.

In order to make a comparison, three performance metrics are defined. The first one is the execution time of the sensor fusion approaches. When we consider the sensor frequency, the two fusion approaches do not have a significant advantage in terms of the execution time compared to the other.

The other performance metrics are mean square error and standard deviation. These metrics, calculated for each state, help to understand the accuracy. From the test results, in the specific regions, it is seen that track-to-track fusion approach improves the accuracy. On the other hand, the low-level fusion approach tracks the target longer than the proposed method when the target vehicle moves away.

In future work, we intend to extend test scenarios including complex traffic conditions such as urban driving to increase the assessment of our proposed method. Also, in order to increase the performance of the track-to-track fusion, the value of the weight factor can be optimized.

REFERENCES

[1] **LiDAR, I.V.** USER‘S MANUAL AND PROGRAMMING GUIDE VLP-16.

[2] SAE Standards News: J3016 automated-driving graphic update, `https://www.sae.org/news/2019/01/sae-updates-j3016-automated-driving-graphic`, accessed: 2020-06-08.

[3] **Rachman, A.S.A.** (2017). 3D-LIDAR Multi Object Tracking for Autonomous Driving: Multi-target Detection and Tracking under Urban Road Uncertainties.

[4] **LiDAR, I.V.**, `https://velodynelidar.com/products/puck/`.

[5] **Kampker, A.**, **Sefati, M.**, **Rachman, A.S.A.**, **Kreisköther, K. and Campoy, P.** (2018). Towards Multi-Object Detection and Tracking in Urban Scenario under Uncertainties.

[6] **Thrun, S.**, **Burgard, W. and Fox, D.** (2005). *Probabilistic robotics.*, Intelligent robotics and autonomous agents, MIT Press.

[7] **Elfring, J.**, **Appeldoorn, R.**, **van den Dries, S. and Kwakkernaat, M.** (2016). Effective World Modeling: Multisensor Data Fusion Methodology for Automated Driving, *Sensors*.

[8] **Schneider, S.**, **Himmelsbach, M.**, **Luettel, T. and Wuensche, H.** (2010). Fusing vision and LIDAR - Synchronization, correction and occlusion reasoning, *2010 IEEE Intelligent Vehicles Symposium*, pp.388–393.

[9] **Mahlisch, M.**, **Schweiger, R.**, **Ritter, W. and Dietmayer, K.** (2006). Sensorfusion Using Spatio-Temporal Aligned Video and Lidar for Improved Vehicle Detection, *2006 IEEE Intelligent Vehicles Symposium*, pp.424–429.

[10] **Leonard, J.**, **How, J.**, **Teller, S.**, **Berger, M.**, **Campbell, S.**, **Fiore, G.**, **Fletcher, L.**, **Frazzoli, E.**, **Huang, A.**, **Karaman, S.**, **Koch, O.**, **Kuwata, Y.**, **Moore, D.**, **Olson, E.**, **Peters, S.**, **Teo, J.**, **Truax, R.**, **Walter, M.**, **Barrett, D. and Williams, J.** (2008). A Perception-Driven Autonomous Urban Vehicle, *Journal of Field Robotics*, *25*, 727 – 774.

[11] **Peterson, K.**, **Ziglar, J. and Rybski, P.E.** (2008). Fast feature detection and stochastic parameter estimation of road shape using multiple LIDAR, *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.612–619.

[12] **Sualeh, M. and Kim, G.W.** (2019). Dynamic Multi-LiDAR Based Multiple Object Detection and Tracking, *Sensors*, *19*(6), 1474.

[13] **Borgmann, B.**, **Schatz, V.**, **Kieritz, H.**, **Scherer-Klöckling, C.**, **Hebel, M. and Arens, M.** (2018). DATA PROCESSING AND RECORDING USING A VERSATILE MULTI-SENSOR VEHICLE., *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, *4*(1).

[14] **Hajri, H. and Rahal, M.** (2018). Real Time Lidar and Radar High-Level Fusion for Obstacle Detection and Tracking with evaluation on a ground truth.

[15] **Gotzig, H. and Geduld, G.**, (2016). Automotive LIDAR, Springer International Publishing, Cham, pp.405–430, `https://doi.org/10.1007/978-3-319-12352-3_18`.

[16] Robot Truck Upstart Embark Hauls In 30 Million To Take On Waymo And Uber, `https://www.forbes.com/sites/alanohnsman/2018/07/19/robot-truck-upstart-embark-hauls-in-30-million-to-take-on-waymo-uber/#437e41b4aee3`, accessed: 2020-04-11.

[17] **Mertz, C.**, **Navarro-Serment, L.**, **Maclachlan, R.**, **Rybski, P.**, **Steinfeld, A.**, **Suppé, A.**, **Urmson, C.**, **Vandapel, N.**, **Hebert, M.**, **Thorpe, C. and Duggins, D.** (2013). Moving object detection with laser scanners, *J. Field Robot.*, *30*, 17–43.

[18] **Sithole, G. and Vosselman, G.** (2004). Experimental Comparison of Filter Algorithms for Bare-Earth Extraction from Airborne Laser Scanning Point Clouds, *ISPRS Journal of Photogrammetry and Remote Sensing*, *59*, 85–101.

[19] **Axelsson, P.** (2000). DEM generation from laser scanner data using adaptive tin models.

[20] **Saraswathi, S. and Sheela, M.I.** (2014). A comparative study of various clustering algorithms in data mining.

[21] **Feng, C.**, **Taguchi, Y. and Kamat, V.R.** (2014). Fast plane extraction in organized point clouds using agglomerative hierarchical clustering, *2014 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, pp.6218–6225.

[22] **Uppada, S.K.** (2014). Centroid based clustering algorithms—A clarion study.

[23] **Banfield, J.D. and Raftery, A.E.** (1993). Model-based Gaussian and non-Gaussian clustering.

[24] **Ester, M.**, **Kriegel, H.P.**, **Sander, J.**, **Xu, X.** *et al.*. A density-based algorithm for discovering clusters in large spatial databases with noise.

[25] **Wang, C.**, **Ji, M.**, **Wang, J.**, **Wen, W.**, **Li, T. and Sun, Y.** (2019). An improved DBSCAN method for LiDAR data segmentation with automatic Eps estimation, *Sensors*, *19*(1), 172.

[26] **Xu, X.**, **Ester, M.**, **Kriegel, H.P. and Sander, J.** (1998). A distribution-based clustering algorithm for mining in large spatial databases, *Proceedings 14th International Conference on Data Engineering*, IEEE, pp.324–331.

[27] **Barrois, B.**, **Hristova, S.**, **Wohler, C.**, **Kummert, F. and Hermes, C.** (2009). 3D pose estimation of vehicles using a stereo camera, *2009 IEEE Intelligent Vehicles Symposium*, pp.267–272.

[28] **Petrovskaya, A. and Thrun, S.** Model based vehicle tracking for autonomous driving in urban environments, *Proceedings of Robotics: Science and Systems IV, Zurich, Switzerland*, *34*.

[29] **Ćesić, J.**, **Marković, I.**, **Jurić-Kavelj, S. and Petrović, I.** (2014). Detection and tracking of dynamic objects using 3D laser range sensor on a mobile platform, *2014 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, volume 2, IEEE, pp.110–119.

[30] **Yutong Ye**, **Liming Fu and Bijun Li** (2016). Object detection and tracking using multi-layer laser for autonomous urban driving, *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp.259–264.

[31] **Verma, R.** (2017). A Review of Object Detection and Tracking Methods, *International Journal of Advance Engineering and Research Development*, *4*, 569–578.

[32] **Mazor, E.**, **Averbuch, A.**, **Bar-Shalom, Y. and Dayan, J.** (1998). Interacting multiple model methods in target tracking: A survey, *Aerospace and Electronic Systems, IEEE Transactions on*, *34*, 103 – 123.

[33] **Pulford, G.W.** (2005). Taxonomy of multiple target tracking methods, *IEE Proceedings - Radar, Sonar and Navigation*, *152*(5), 291–304.

[34] **Houenou, A.**, **Bonnifait, P.**, **Cherfaoui, V. and Boissou, J.** (2012). A track-to-track association method for automotive perception systems, *2012 IEEE Intelligent Vehicles Symposium*, pp.704–710.

[35] **Aeberhard, M.**, **Rauch, S.**, **Bahram, M.**, **Tanzmeister, G.**, **Thomas, J.**, **Pilat, Y.**, **Homm, F.**, **Huber, W. and Kaempchen, N.** (2015). Experience, Results and Lessons Learned from Automated Driving on Germany's Highways, *IEEE Intelligent Transportation Systems Magazine*, *7*, 42–57.

[36] **Houenou, A.**, **Bonnifait, P.**, **Berge-Cherfaoui, V. and Boissou, J.F.** (2012). A track-to-track association method for automotive perception systems, *2012 IEEE Intelligent Vehicles Symposium*, 704–710.

[37] **Konstantinova, P.**, **Udvarev, A. and Semerdjiev, T.** (2003). A Study of a Target Tracking Algorithm Using Global Nearest Neighbor Approach 1.

[38] **Bar-Shalom, Y.**, **Daum, F. and Huang, J.** (2010). The probabilistic data association filter, *Control Systems, IEEE*, *29*, 82 – 100.

[39] **Mclachlan, G.** (1999). Mahalanobis Distance, *Resonance*, *4*, 20–26.

[40] **Fortmann, T.E.**, **Bar-Shalom, Y. and Scheffe, M.** (1980). Multi-target tracking using joint probabilistic data association, *1980 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, pp.807–812.

[41] **Schreier, M.**, **Willert, V. and Adamy, J.** (2016). Compact Representation of Dynamic Driving Environments for ADAS by Parametric Free Space and Dynamic Object Maps, *IEEE Transactions on Intelligent Transportation Systems*, *17*(2), 367–384.

[42] **Schreier, M.**, **Willert, V. and Adamy, J.** (2016). Compact Representation of Dynamic Driving Environments for ADAS by Parametric Free Space and Dynamic Object Maps, *IEEE Transactions on Intelligent Transportation Systems*, *17*, 367–384.

[43] **Julier, S.J. and Uhlmann, J.K.** (1997). A non-divergent estimation algorithm in the presence of unknown correlations, *Proceedings of the 1997 American Control Conference (Cat. No.97CH36041)*, volume 4, pp.2369–2373 vol.4.

APPENDICES

APPENDIX A : Figures

APPENDIX A

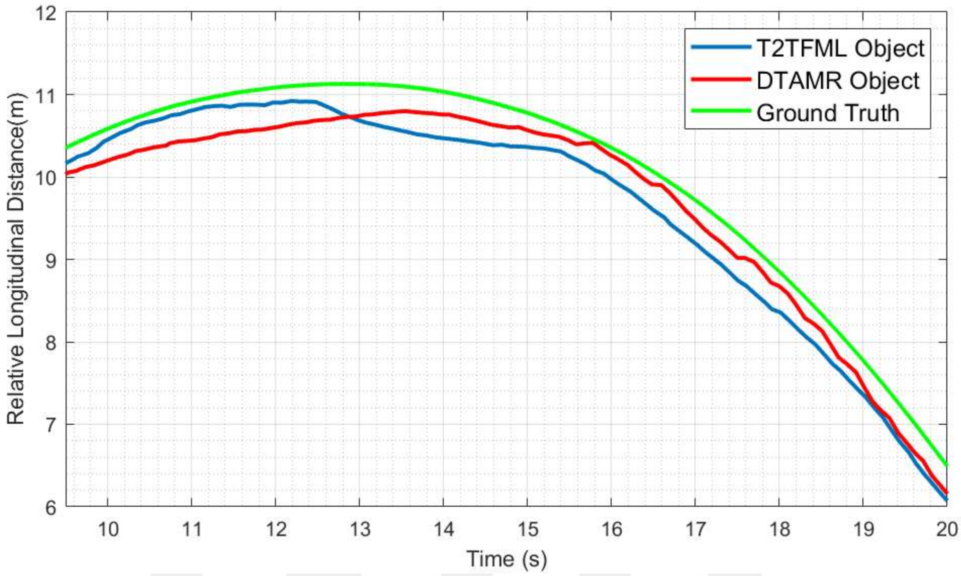

Figure A.1 : Relative longitudinal position of the object with ground truth data for Scenario 3.

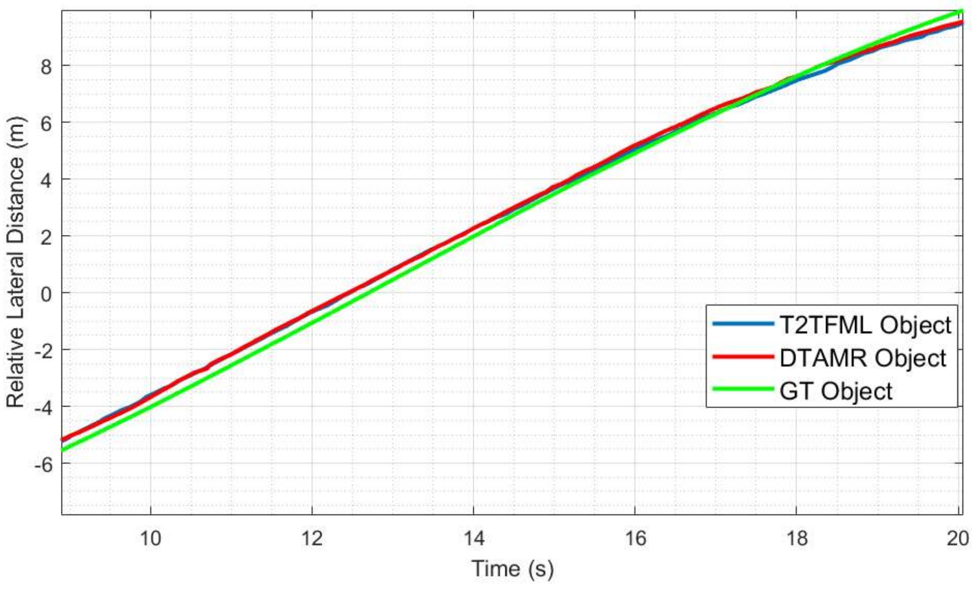

Figure A.2 : Relative lateral position of the object with ground truth data for Scenario 3.

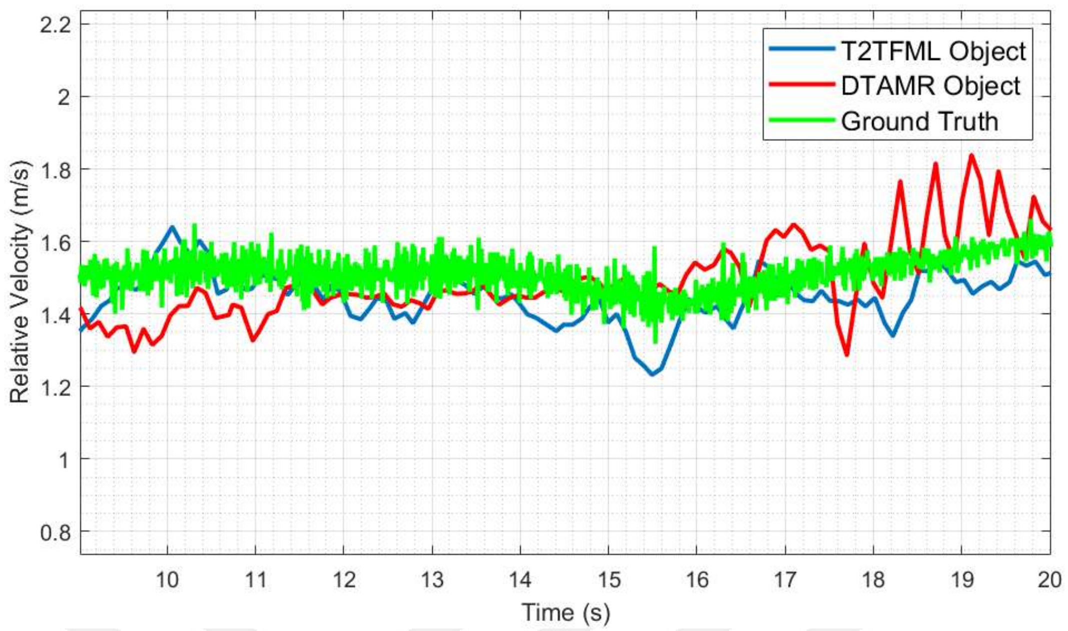

Figure A.3 : Relative velocity of the object with ground truth data for Scenario 3.

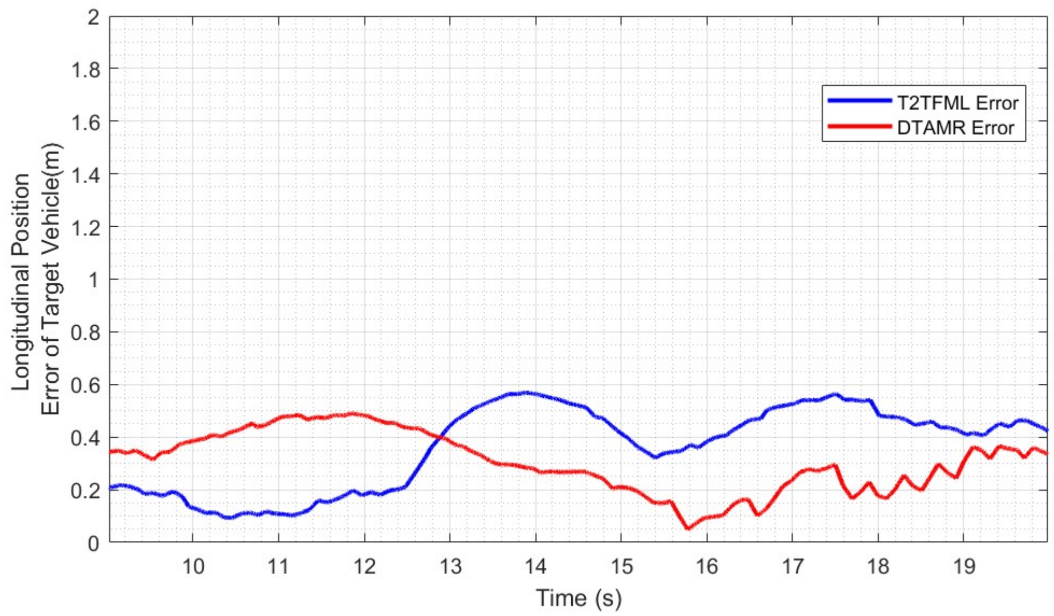

Figure A.4 : Absolute error of estimated object longitudinal position state for Scenario 3.

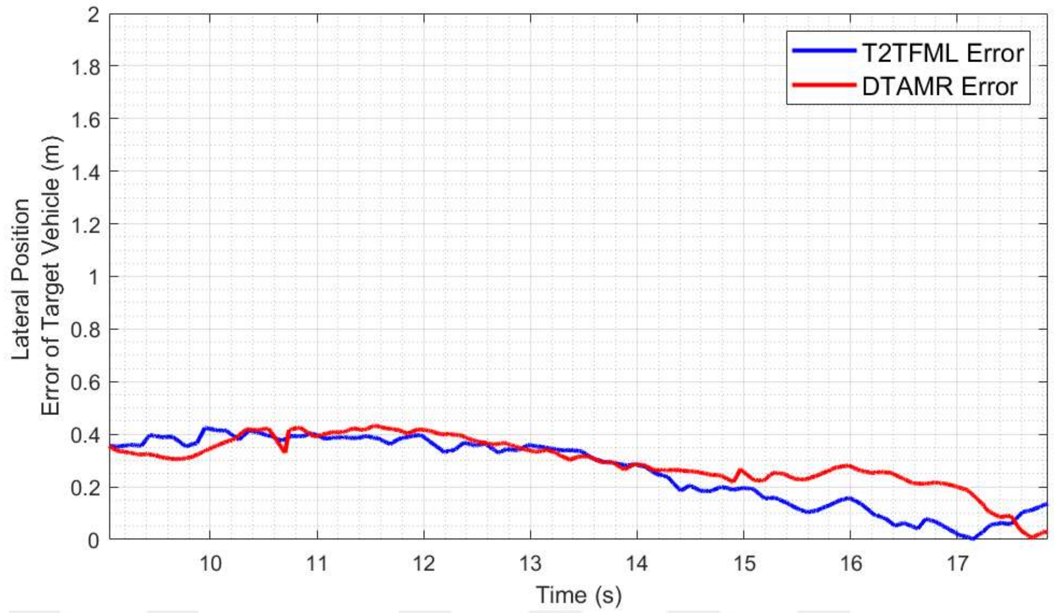

Figure A.5 : Absolute error of estimated object lateral position state for Scenario 3.

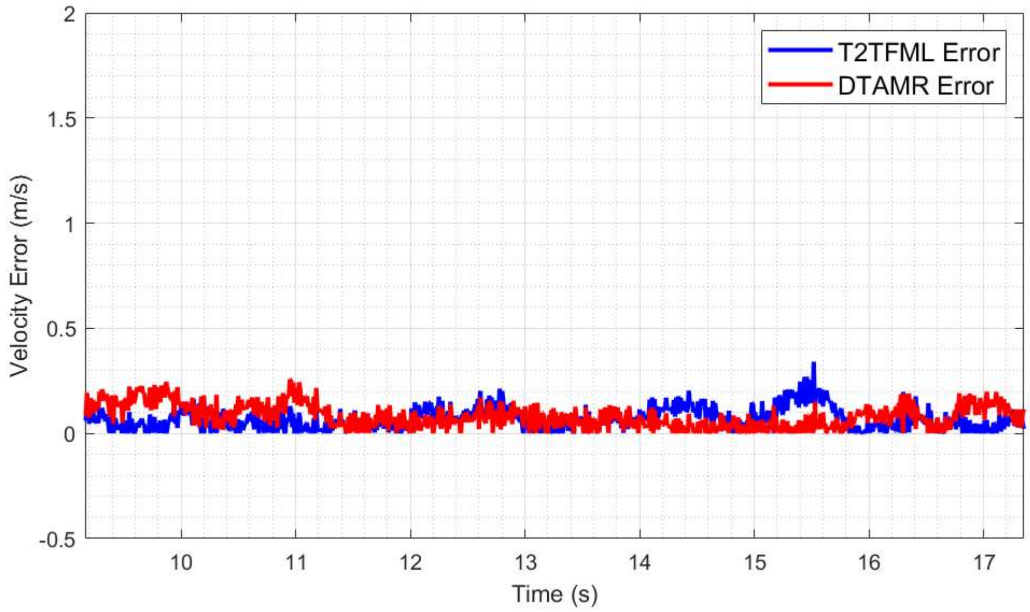

Figure A.6 : Absolute error of estimated object velocity state for Scenario 3.

CURRICULUM VITAE

Name Surname : **Elif AKSU TAŞDELEN**

Place and Date of Birth : **12.09.1992 – Pendik/ISTANBUL**

E-Mail : **elifaksutasdelen@gmail.com**

EDUCATION :

- **B.Sc.** : 2015, Middle East Technical University, Electical and Electronics Engineering

PROFESSIONAL EXPERİENCE AND REWARDS :

- **2018 - Present** : Autonomous Vehicles Engineer at FORD OTOSAN
- **2017 - 2018** : Systems Engineer at AVL Research and Development Turkey
- **2015 - 2016** : System Engineer at ROKETSAN

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Buluttekin H., Aksu E., Bhanuprakash, A.,** Comparative Study of Data Association Approaches for Multiple Vehicle Tracking, *System Theory, Control and Computing (ICSTCC), 2017. 21st International Conference, 19-21 Oct. 2017, Sinai, Romania*