ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

MULTI-SENSOR LANE TRACKING AND LANE DEPARTURE WARNING SYSTEM DESIGN

M.Sc. THESIS

Barış ÖZCAN

Department of Mechatronics Engineering

Mechatronics Engineering Programme

MAY 2014

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

MULTI-SENSOR LANE TRACKING AND LANE DEPARTURE WARNING SYSTEM DESIGN

M.Sc. THESIS

Barış ÖZCAN
(518111004)

Department of Mechatronics Engineering

Mechatronics Engineering Programme

Thesis Advisor: Asst. Prof. Dr. Pınar BOYRAZ

MAY 2014

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ÇOK-SENSÖRLÜ ŞERİT TAKİP VE ŞERİTTEN AYRILMA UYARI SİSTEMİ TASARIMI

YÜKSEK LİSANS TEZİ

Barış ÖZCAN
(518111004)

Mekatronik Mühendisliği Anabilim Dalı

Mekatronik Mühendisliği Programı

Tez Danışmanı: Yrd. Doç. Dr. Pınar BOYRAZ

MAYIS 2014

Barış Özcan, a **M.Sc.** student of ITU **Graduate School of Mechatronics Engineering** student ID **518111004**, successfully defended the **thesis** entitled "**MULTI-SENSOR LANE TRACKING AND LANE DEPARTURE WARNING SYSTEM DESIGN**", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Asst. Prof. Dr. Pınar BOYRAZ** İstanbul Technical University	.....
Jury Members :	**Prof. Dr. Şeniz Ertuğrul** İstanbul Technical University	.....
	Asst. Prof. Dr. Figen Özen Haliç University	.....

Date of Submission : 5 May 2014
Date of Defense : 29 May 2014

To the dreamers and those who put faith in dreams as in the only realities,

FOREWORD

Before all, I would like to thank my thesis advisor Ass. Prof. Dr. Pınar BOYRAZ for giving me an opportunity to work under her guidance. Also, I would like to thank to OTOKAR and Ministry of Science, Industry and Technology for financing my thesis. Additionally, I would like to thank to my family and to those who have always supported me throughout my life. Particularly Cihat Bora YİĞİT and Ekim YURTSEVER motivated me with their sarcastic comments on my pschopathology during rough times. Lastly, I would like to give my sincere gratitudes to Ziya ERCAN and Ertuğrul BAYRAKTAR for their encouragement.

May 2014

Barış ÖZCAN
Mechatronics Engineer

TABLE OF CONTENTS

Page

ABBREVIATIONS

ADAS : Advanced Driver Assistance Systems
LDW : Lane Driver Warning
PNG : Positive Negative Gradients
EKF : Extended Kalman Filter
ROI : Region of Interest
IPM : Inverse Perspective Mapping
SLAM : Simultaneous Localization and Mapping

LIST OF FIGURES

Page

MULTI-SENSOR LANE TRACKINF AND LANE DEPARTURE WARNING SYSTEM DESIGN

SUMMARY

Reports by World Health Organization claim that a large portion of the traffic accidents caused by human errors. In order to avoid these accidents advanced driver assistance systems (ADAS) such as lane departure warning systems are widely researched in the recent decades. Lane departure warning systems utilize computer vision algorithms in order to find and track lanes. Different algorithms for edge detection, outlier removal, road models and model fitting methods are used with various combinations of tracking algorithms such as Kalman filter or particle filter. However, most common approach in the literature is to track the vehicle inside a lane and assume that vehicle is stationary and lanes are moving which is practically not true.

In this thesis, a lane tracking system approaching the problem as a mapping problem meaning that the vehicle is moving in a previously unknown environment is developed. The proposed algorithm can be divided into two main sections, the image processing module and the tracking algorithm. In the image, processing module an adaptive region of interest (ROI) creation by using the estimations provided by the tracking algorithm is introduced. In addition, a simple solution for the perspective problem while using the steerable filters are presented. The orientation of the steerable filter is vital for edge detection and in order to obtain results that are more robust, the orientation of the steerable filter is developed to be adaptive according to the estimates from the tracking algorithm. Outliers in the obtained image after the filtering process Hough transform is applied to each segment to remove the outliers. Than a first order polynomial is fit the do features on the image in least squares sense. Extended Kalman Filter (EKF) is used in order to track the found lanes via the image processing module. Tracking is applied approaching the problem as a mapping and localization problem. However, the localization problem is simplified into localizing the vehicle laterally on the road according to the initial position. Tracking algorithm can be divided into two main parts, the motion model and the measurement model. While developing the motion model only the lateral position of the vehicle is considered. The vehicle is modelled with regard to the constant velocity model in the literature. Each tracked lane is represented with five control points at pre-determined distances ahead of the vehicle in the motion model. Motion of the control points are modelled as if they have a constant position according to the world plane. In the measurement model, lateral position of the each control point on the world plane is regarded as measurements. However, since the control points are measured on image plane the measurement model is developed as the projection of the control points from the image plane to world plane. In order to achieve this, pinhole camera model is used to project the control points from the image plane to camera frame. Next, with an assumption of constant yaw, pitch and roll between the camera frame and the world plane a rotation

matrix and translation vector is defined to project the control points from the camera frame to world plane. Additionally an altered version of the proposed tracking algorithm is presented. This altered version utilizes fusion of IMU and computer vision. The previously proposed model is altered with two new states, which are yaw and the yaw rate of the camera frame. Adding these two states discards the assumption of constant yaw between the world plane and the camera frame. Two new states are also modelled with regard to constant velocity model. In the measurement model, measurements of yaw rate provided by the IMU is added to the model.

In order to make experiments with the algorithms, synchronized data of camera and IMU is collected with a test vehicle. The collected data is synchronized and stored in MATLAB. Experiments are made offline on MATLAB with the previously collected data. Tracking results of the control points are given along with the measurements on the image plane during various scenarios such as when no measurement is received or during curvature roads. Localization results of the vehicle is tested with a test similar to the loop closure test, which is a commonly used test in SLAM algorithms. The results of the loop closure tests claim that the tracking algorithm can successfully track the lateral position of the vehicle with an acceptable error.

In this thesis, the main objective is to propose an alternative lane tracking system capable laterally localizing the vehicle and map the lanes according to the initial position. Additionally this proposed system provides a very versatile basis for sensor fusion with various sensors such as GPS or LiDAR.

ÇOK–SENSÖRLÜ ŞERİT TAKİP VE ŞERİTTEN AYRILMA UYARI SİSTEMİ TASARIMI

ÖZET

2010 yılındaki raporlara göre trafikte 1 milyarın üstünde taşıt bulunmakta ve günlük hayatta hem şehir içi hem şehir dışı ulaşım için çok yoğun bir şekilde taşıt kullanılıyor. Bu durum göz önüne alındığında, kullandığımız taşıtların güvenliği can güvenliğimiz için önemli bir role sahip. Dünya Sağlık Kuruluşu (WHO) tarafından 2013 yılında yayınlanan raporda trafik kazalarının insan ölümlerinin nedenleri sıralamasında sekizinci sırada olduğu ifade edilmiştir. Aynı raporda yapılan araştırmalarda bu trafik kazalarının %90'ı sürücü hatalarından kaynaklandığı sonucuna varılmıştır. Bu sürücü kazalarını engellemek amacı ile araç güvenliği sistemlerini zorunlu kılan yasa tasarıları hazırlayarak hükümetler araç güvenliği sistemlerini son yıllarda daha yaygın hale getirmeye çalışmaktadır.

Araç güveliği sistemleri aktif ve pasif olarak ikiye ayrılır, aktif araç güvenlik sistemlerinin tasarım amacı trafik kazalarını engellemek amacıyla tasarlanmış sistemlerdir. Diğer yanda pasif araç güvenliği sistemleri ise, trafik kazalarının etkilerini azaltmak üzerine tasarlanmış sistemlerdir. Aktif araç güvenlik sistemlerine kilitlenme karşıtı frenleme sistemi, çekiş kontrol sistemleri veya şerit takip sistemleri örnek olarak verilebilir. Öte yandan emniyet kemeri, hava yastığı ve koltuk başlıkları pasif araç güvenliğindeki ilk akla gelen örneklerden bazılarıdır.

Bu tez çalışmasında, IMU ile desteklenmiş yapay görü ile şerit takip sistemi tasarlanmış ve test edilmiştir. Şerit takip sistemleri bir çok alt algoritmayı yapısında bulundurması nedeniyle geçmişte bu alanda yapılan çalışmalar beş başlık altında anlatılabilir. Bu başlıklar şu şekilde isimlendirilebilir , yapay görüde şeritleri bulma yöntemleri, şerit modelleri, şeritlere ait olmayan örnekleri ayırma, takip ve sisteme destek olacak yapay görü harici algılayıcıların eklenmesi.

Kameradan elde edilen görüntüler genelde, görüntüdeki köşe veya kenarları ön plana çıkaracak ,'steerable' filtre veya 'sobel' filtresi gibi filterelerden geçirilir. Bu filterelerden geçen görüntü belli bir eşik değerine göre kestrim uygulanıp iki rakamlı görüntüye yani sadece birlerden ve sıfırlardan oluşan bir görüntüye çevrilir. Bu filterden bazıları , örneğin 'steerable' filtre sabit bir şerit işareti genişliği için doğru sonuclar verebilir. Fakat kamera aracın önündeki yola baktığı için perspektif sebebi ile şeritler araçtan uzaklaştıkça şerit işareti genişliği azalır. Bu problemi aşmak için litaratürde, ters perpektif haritalama uygulaması önerilmiştir. Bu haritalama methodu sayesinde görüntü kuş bakışına çevrilir ve perspektif problemi ortadan kalkar.

Şeritin kullanılan modeli ile , şeritlere ait olmayan örnekleri ayırma işlemleri birbirleri ile doğrudan bağlantılıdır. Şerit veya yol modeli , sistemin tahmini çalışma koşulları göz önüne alınarak belirmesi gerekir. Örneğin, yüksek hızlarda çalışacak bir şerit takip sistemi için şerit veya yol modelinin daha düşük hızlar için tasarlanmış sistemlere göre aracın önündeki daha uzun bir mesafede yolu doğru temsil edebilmesi gerekir. Aynı

zamanda yol modelleri işlem gücü gereksinimini doğrudan belirleyen bir faktör olduğu için çok ayrıntılı bir model de gerçek zamanlı çalışma durumunda problemler yaratabilir. Şeritlere ait olmayan örnekleri ayırma veya filtreleme işleminde elde edilen veriye önceden blirlenmiş bir model uydurma işlemi yapay görü sisteminin şerit bulma aşamasındaki son adımıdır. Bu adım sonucunda belirlenmiş bir modelin bilinmeyen katsayıları elde edilen verilere göre tahmin edilir. Litaratürde bu işlem için , RANSAC, en az kareler veya metropolis algoritması gibi yöntemler kullanılmıştır.

Yapay görü sayesinde bulunan şeritler hem sadece yapay görü sistemi bünyesinde hem de başka algılayıcılar ile birlikte kullanılarak takip edilebilir. Şerit takibi için en sık kullanılan yöntemlerden doğrusal sistemler için bir uyarlamalı kestrim algoritması olan Kalman filtresidir. Fakat, şerit takip sistemleri ancak gerçek ile oldukça çelişen ciddi varsayımlar yapılarak doğrusal bir sistem haline getirilebilir. Bu nedenle Kalman filtresi temel alınarak doğrusal olmayan sistemler için geliştirilmiş olan genişletilmiş Kalman filtresi ve yine doğrusal olmayan sistemler için uygun olan parçacık filtresi litaratürde kullanılmıştır.

Litaratürde şerit takip sistemleri genellikle , aracın şerit içindeki hareketini takip etmek üzere tasarlanmışır. Ayrıca, görüntü üzerinden yola çıkılması sebebi ile, yapay görü ve şerit takip sisteminde araç sabit kabul edilip şeritler hareketli kabul edilmiştir ki bu pratikte kesinlikle yanlıştır. Bu tez çalışmasında şerit takip sisteminin çalıştığı taşıtın önceden bilinmeyen bir ortamda çalıştığı var sayılarak probleme haritalama mantığı ile yaklaşılmıştır. Bu yaklaşım kapsamında litaratürden farklı olarak, şeritlerin hareketli aracın sabit olma kabulu yerine şeritlerin dünya düzleminde sabit fakat aracın hareketli olduğu kabul edilmiştir. Ayrıca ortaya çıkarılan sistem başka algılayıcılar ile kolayca etkileşim halinde modifiye edilebilecek şekilde tasarlanmış ve IMU ile birlikte çalışması incelenmiştir.

Sunulan sistem yapay görü modülü ve şerit takip sistemi olarak iki parcaya bölünmüştür. Yapay görü sisteminde takip algoritmasından elde edilen şeritlerin tahmini konumlarının olduğu bölgede kameradan alınan görüntü kırpılarak filtereleme işlemi için hazırlanır. Ardından perspektif problemini aşmak için görüntü beş parçaya ayırıp her parca icin önceden belirlenmiş boyuttaki ‘steerable’ filtreler sayesinde şeritleri ait yerlerde 1 ait olmayan yerlerde 0 değerlerine sahip görüntü Canny köşe bulma algoritmasına beslenir. Görüntüdeki her bölümde Hough transformu uygulanarak şeritlere ait çizgiler bulunduktan her parça için bulunan çizgiler en az kareler yöntemine beslenerek şeritlere ait birinci derecen polinom modelinin katsayıları bulunmuştur.

Takip algoritmasında genişletilmiş Kalman filtresi kullanılmıştır ve iki bölümden oluşmaktadır, ölçüm modeli ve sistem modeli. Takip algoritması aracın ve şeritlerin sadece yatay eksendeki pozisyonlarının takip etmek üzere tasarlanmıştır. İşlem gücünü sadece bir noktanın yatay eksende takibine indirgemek için her şeriti temsil edecek beş kontrol noktası kullanılmıştır. Modelde araç ile kameranın koordinat düzlemi eşlenik kabul edilip dünya koordinat düzlemi ile arasında sabit bir rotasyon olduğu kabul edilmiştir. Modelde kullanılan durum vektöründe aracın yatay pozisyonu ve yatay hızının yanısıra her kontrol noktasının yatay pozisyonu da eklenmiştir. Araç sabit hız modeli kontol noktaları için sabit pozisyon modeli baz alınarak modellenmiştir.

Ölçüm modelinde ise görüntü düzlemindeki kontrol noktalarının dünya koordinat düzlemine iz düşümü ölçüm modeli olarak alınmıştır. Bu işlem iki aşamalıdır, önce görüntü düzleminden kamera koordinat düzlemine iz düşümü alınmasının ardından kamera koordinat düzleminden dünya koordinat düzlemine iz düşümü alınır. Analitik

olarak görüntü ile kamera koordinat düzlemi arasındaki iz düşümünü tanımlamak için iğne deliği kamera modeli kullanılmıştır. Kamera koordinat düzlemi ile dünya düzlemi arasında ise, Euler açıları ile elde edilen rotasyon matrisi ve öteleme vektörü kullanılmıştır. Önerilen sisteme IMU verisini ekleyebilme amacı ile önceki modele iki yeni durum eklenmiştir. Bu iki yeni durum yalpa açısı ve yalpanın açısal hızı olarak belirlenmiştir. IMU'nun ilk aşama olarak sadece ciroskop verisi ölçüm modeline eklenmiştir.

Önerilen algoritma önceden bir test aracı ile toplanılmış senkronize IMU ve kamera verisi üzerinde bilgisayar üzerinde test edilmiştir. Veri toplama aracındaki sistemdeki senkronizasyon MATLAB üzerinden IMU'nun yapısında var olan ara bellek kullanılarak başarılmıştır. IMU 90Hz kamera is 9Hz frekansında senkronize bir şekilde ölçüm almıştır. Algoritma toplanan MATLAB aracılığı ile önceden toplanan veri üzerinde farklı senaryolar için test edilmiştir. Sonuclar kısmında öncelikle görüntü düzlemindeki kontrol noktalarının takip algoritması kestrimleri ve yapay görü modülünden alınan ölçümleri sunulmuştur. Ayrıca haritalama algoritmalarında sık kullanılan bir test olan 'loop closure' testine benzer bir test uygulanmıştır. Bu test bir haritalama algoritmasının bir ortamda dolaştıktan sonra tekrar aynı yere getirerek uygulanır. Bu duruma benzer olarak önceden toplanan veri üzerinde aracın yatay olarak başlangıç noktasına geri geldiği bir video sekansı üzerinde algoritma test edilip sonuçlar verilmiştir. Sonuçlar algoritmanın aracın aynı noktaya geri geldiğini kabul edilebilir bir hata ile tahmin ettiğini göstermiştir.

Önerilen algoritmanın ileriki çalışmaları bir kaç farklı açıdan ele alınabilir. Yapay görü modülünde kullanılan şerit modeli ve model katsayıları tahmin eden algoritmalar için litaratürdeki farklı yöntemler uygulanabilir. Ayrıca şerit takip sistemi için ise, daha fazla algılayıcıdan elde edilen veri bir arada kullanılacak şekilde sistem geliştirilebilir veya genişletilmiş Kalman filtresi yerine parçacık filtresi uygulanabilir.

1. INTRODUCTION

The number of active commercial and personal vehicles in traffic in all around have surpassed 1 billion-unit mark in 2010 according to the report of Ward, which equates roughly to a ratio of 1:6.75 vehicles to people among a world population of 6.9 billion [1]. The number of vehicles refer to the number of cars, light-, medium- and heavy-duty trucks and busses registered worldwide, excluding the off-road vehicles. Vehicles are used in personal and public transportations, logistics and many more areas. As vehicles become a vital part of our daily lives, preventing vehicle accidents became a priority all around the world.

Global Status Report on Road Safety of 2013 published by World Health Organization (WHO) [2] claim that road traffic injuries are estimated to be the eighth leading cause of death globally also is the leading cause of death for young people aged between 15-29 years. If an estimation to be made from current trends, by 2030 road traffic deaths will become the fifth leading cause death unless any action is taken. Reports also claim that 1.2 million people are killed in transport accidents each year; moreover, 90% of those are caused by human errors alone. Some examples for leading causes of human errors are drowsiness, driver distraction and intoxicated driving.

In recent years, governments start to implement road safety strategies in order to prevent vehicle accidents. This road safety strategy includes laws against driver behaviors that endanger road safety and technical requirements for vehicles that assist drivers to prevent accidents. An example for this technical requirement regulation particularly on Lane Departure Warning Systems (LDWS) is European Unions (EU) commission regulation no 351/2012 [3].

The rising trend in traffic accidents in the recent years led to a wider and more through research on vehicle safety systems in the recent decades. One of the first formal academic studies into improving vehicle safety was by Cornell Aeronautical Labs of Buffalo, New York, which pointed out the importance of seat belts and padded

dashboards. Vehicle Safety Systems can be divided into two main streams, Active and Passive Vehicle Safety Systems, which are to be explained herein.

First, Passive Vehicle Safety is defined as the systems that reduce the effect of the accident on the individuals during or after the accident. Some examples for Passive Safety Systems can be listed as crumple zones, seatbelts or headrests. For example the crumple zone is an on-purpose crush space which is a physical feature of the car that absorbs the energy with a controlled deformation during a crash, whereas seat belts that keep the occupants in their place when an accident happen. Furthermore, headrests can be considered as passive safety systems with mitigation effect, which because they are designed to prevent whiplash injuries in people by supporting the head during emergency stops or collisions.

Active Vehicle Safety (AVS) Systems are designed to prevent accidents from happening. Until recent years. Active Vehicle Systems was referred to non – complex safety precautions such as good visibility or low interior noise levels. However, in the last two decades, complex Active Vehicle Safety Systems became more prominent. For example, anti-lock breaking system (ABS) in an AVS system that allows the wheels on a motor vehicle to maintain tractive contact with to road surface [4]. Furthermore, Electronic Stability Control (ESC) system is an AVS system that applies breaking automatically to help steering by detecting and reducing lateral skidding [5]. Furthermore, Traction Control Systems (TCS) which is similar to ESC however TCS consider vertical skidding instead of lateral skidding [6][7].

Among Active Vehicle Safety Systems, Advanced Driver Assistance Systems (ADAS) are also included and they try to include the human driver in the loop. ADAS is to act as a virtual co-pilot for the driver, assisting the driving process. Some examples of these systems can be given as follows: Adaptive Cruise Control System adjusts the speed to maintain the safety distance according to travelling speed with the vehicle ahead [8]. Adaptive Light Control (ALC) is designed to improve nighttime safety. ALC controls the headlamp illumination according to the current driving situation and environment [9]. Another common ADAS nowadays is Blind Spot Information System (BLIS), which is developed by Volvo [10]. Vehicles have blind spots from drivers' point of view; even though these blind spots can be removed with proper side view mirror adjustment it is commonly overlooked. BLIS warns the driver if a car enters the blind spots of the Vehicle. Driver Drowsiness Detection is another

example for ADAS. Some examples from companies that have Driver Drowsiness Detection System are as follows ; Ford's Driver Alert , Mercedes-Benz's Attention Assist, Volkswagen's Fatigue Detection System and Volvo's Driver Alert Control [11]. Finally, Lane Departure Warning Systems (LDWS), which is the focus of this thesis among the ADAS systems. LDWS warns the driver when the vehicle begin to move out of its lane. LDWS' are explained particularly in section 1.2.

As explained previously, ADAS are widely researched topic in the recent decade, with its complexity, human machine interface problems and a transition basis to semi-autonomous vehicles ADAS researches yields results to many solutions for futuristic Vehicles.

1.1 Purpose of Thesis

With the increasing death toll due to traffic accidents in the recent decades there is a strong demand in active vehicle safety systems from the industry, moreover governments started to mandate inclusion of this systems in vehicles as explained previously. Despite there are various LDWS in the literature and industry, most of them are only capable of working in well-maintained roads and better weather conditions. Moreover, almost every application involves either only cameras or concept cars that possess high end and expensive equipment. These concept cars are not feasible for the industry at the moment due to expensive sensors such as LIDAR or stereo cameras. Additionally, methods and algorithms are used in LDWS are made particularly for the car that is the LDWS is to be applied. Regarding to these problems, the purpose of this thesis is to construct a more robust LDWS system that is car independent using fusion of image processing and IMU.

1.2 Literature Review

Majority of the LDW systems in the literature use computer vision for lane detection and tracking. The computer vision system is, even though it is not common, supported with peripheral sensors in order to reduce the limitations of LDW because, computer vision systems are vulnerable to noise in the environment. Some of the supporting sensors usually used in literature are, Global Positioning System (GPS), and Light Detection and Ranging (LIDAR).

Computer vision system of a LDWS often employs an algorithm with three main steps; lane feature detection, lane model parameter estimation and tracking. There are various combinations of lane feature detection, lane modelling. All these steps can be performed using various combinations of several algorithms for each. In this part for the sake of clarity, literature is overviewed five separate parts; lane feature extraction, lane modelling, post processing, sensor fusion and tracking.

Lane feature extraction is the core of the LDW system. Lane features are extracted via various edge finding filters applied on the image. These filters exploit the geometric and photometric features such as brightness of the lanes in order to detect them. Each of these filters have their particular drawbacks and benefits. In addition, the feature extractor should be chosen according to the working environment of the LDWS and it might not be necessary to extract all the possible variations of the lanes.

One of these methods is called positive – negative gradients extractor (PNG). The method is mainly based on the width of the feature. PNG extractor algorithm first computes gradients on the image and thresholds the gradients and then searches for a pair of positive and negative gradients within a range [12]. PNG has a width constraint on the feature by definition. This constraint is handled with two approaches and both of them rely on a planar road assumption and the knowledge of the horizon line [13].

The first approach used in PNG depends on the Inverse Perspective Mapping (IPM) to re-create the top-view image of the road to rule out the width change on the lanes due to perspective of the image [14]. Second method assumes that the marking width decreases linearly as the lane goes from the bottom of the image to horizon line and has a zero width on the horizon line [15]. Second algorithm is preferred because of its fast response in practice.

In contrast to PNG steerable filters considers continuity of the lanes assuming they are stripes on the road. This is a big advantage when the lanes are eroded or have less visible parts on the image due to various reasons such as illumination or poor road marking. Steerable filters is applied for lane detection in one recent study [16]. Steerable filters which are used for lane feature extraction has a base of second derivation of 2D Gaussians. Benefit of this approach is that this filter is a linear combination of three base filters, which enables it to be computed separately at the same time. With exploiting this advantage, the filter can be applied to the image for

any orientation. As PNG, steerable filters also have a feature width constraint; steerable filters can only extract features with constant width. In most of the applications of steerable filters, this constraint is overcome with IPM transformation [17].

The top-hat filter on the other hand proposed in [13] is to extract lanes in a large width range; however, top-hat filter only considers vertical lines on the image while resolving the IPM requirement. Top- hat filter is applied at many scales to the image in order to overcome the width constraint when IPM is not applied.

Roads or lanes in particular are modelled in the literature. Modelling greatly effects, the system performance as it helps the outlier removal. Road and lane models should be chosen considering the working environment of LDWS. For example, for highway scenarios, due to high speed of vehicles accurate road modelling up to 30m or more may be required therefore a more detailed road model would be more suitable. On the other hand, on high-speed roads such as highways the road can be assumed to have relatively less curvature. There are many types of road and lane modelling in the literature. In the proposed thesis lanes are modelled rather than the road, therefore in the following paragraphs most common lane models are discussed.

The simplest lane model is straight lines [18]. Although it is an oversimplified assumption by segmenting the image in to parts and assuming that the lane is straight inside these segments, straight-line model becomes acceptable. Moreover, in highways and high speeds the lanes are relatively straight to a distance from the vehicle.

As straight lines cannot model the curved lanes, third order parabolas or splines [19] are used to approach curved roads/lanes more accurately. In [19] two lane markings are considered as symmetric and converging to the same vanishing point. Using this assumption two lanes are modelled together and they reduced the parameters on the model to be estimated.

In order to avoid abrupt changes in steering angle while transition from straight to circular road section and vice versa, a special type of curve called clothoids is commonly used in highway construction [20]. An example of lane tracking with clothoid models can be seen in a recent study [21].

Extracted features regarding to the lanes may contain outliers due to noise in the environment. Therefore, outlier removal or post processing is necessary. These

estimates can be regarded as the a priori estimates of the parameters of the model that is used.

One such post-processing method employs Hough transform [22]. The original Hough Transform is used for straight lines and circles however, generalized Hough transform that is proposed in [23] can be used for arbitrary shapes even parallel lines in the future. Another method that is commonly used in computer vision systems is random sample and consensus (RANSAC) proposed which is proposed in [24]. RANSAC is actively used in [25] to fit a model to the extracted data.

Supporting computer vision system with LIDAR, GPS is the most common approach in the literature. In [16], GPS and LIDAR is used to support the proposed LDWS. In [26] LIDAR and GPS is also used however, rather than a monocular camera a stereo and Omni-directional camera is used. Another interesting application in the literature can be seen in [27]. Instead of using LIDAR and GPS as in conventional approach, accelerometer and gyroscope is used to support computer vision system.

Algorithms tracking the lanes can achieve this using only the outputs of computer vision system or the very same algorithms with some modification can track the lanes with a fusion of sensors. In [28], Kalman filter is applied to track the lanes. Since Kalman Filter is applicable to linear systems, for non- linear models Extended Kalman Filter (EKF) is used [29]. Another trending method for tracking is Particle Filters [30], [31], [32]. Computation cost of these algorithms and the nature of the problem is the main factor for the selection of the tracking algorithm. While Particle Filter is easy to implement in sensor fusion, the computational complexity increases exponentially as the number of dimensions increase in the state space. On the other hand, Kalman filter's complexity does not increase exponentially due to increase in the dimensions of state space; however, it may not be suitable for systems having nonlinear behavior and abrupt changes.

Most of the lane tracking methods does not include vehicle or camera model inside model for Kalman Filtering and even if the vehicle model is included, vehicle movement models do not use a constant velocity model. Moreover, almost each one of LDW systems track the vehicle inside a lane or according to lanes not on the whole road the vehicle is travelling.

2. IMAGE PROCESSING

In order to track the lanes and assess the deviation of the vehicle from the lanes, the system must first detect the lanes on the image. In this section, image processing part explains the processing made on one frame alone to detect the lanes on the current image. The processing flow chart is illustrated on Figure 2.1. All the steps in the flow chart is explained in this section. Input of the flow chart is a greyscale image and the output refers to the lane model parameters. Image processing part is explained in three sections. First ROI extraction, Gaussian blurring process and segmentation is explained. Secondly, lane features extraction is explained which refers to the steerable filters, thresholding and Canny edge detection inside the flow chart. Lastly, outlier removal and model parameter estimation is explained. Outlier removal and model parameter estimation is indicated as name of the algorithms used, Hough Transform and Least Squares respectively.

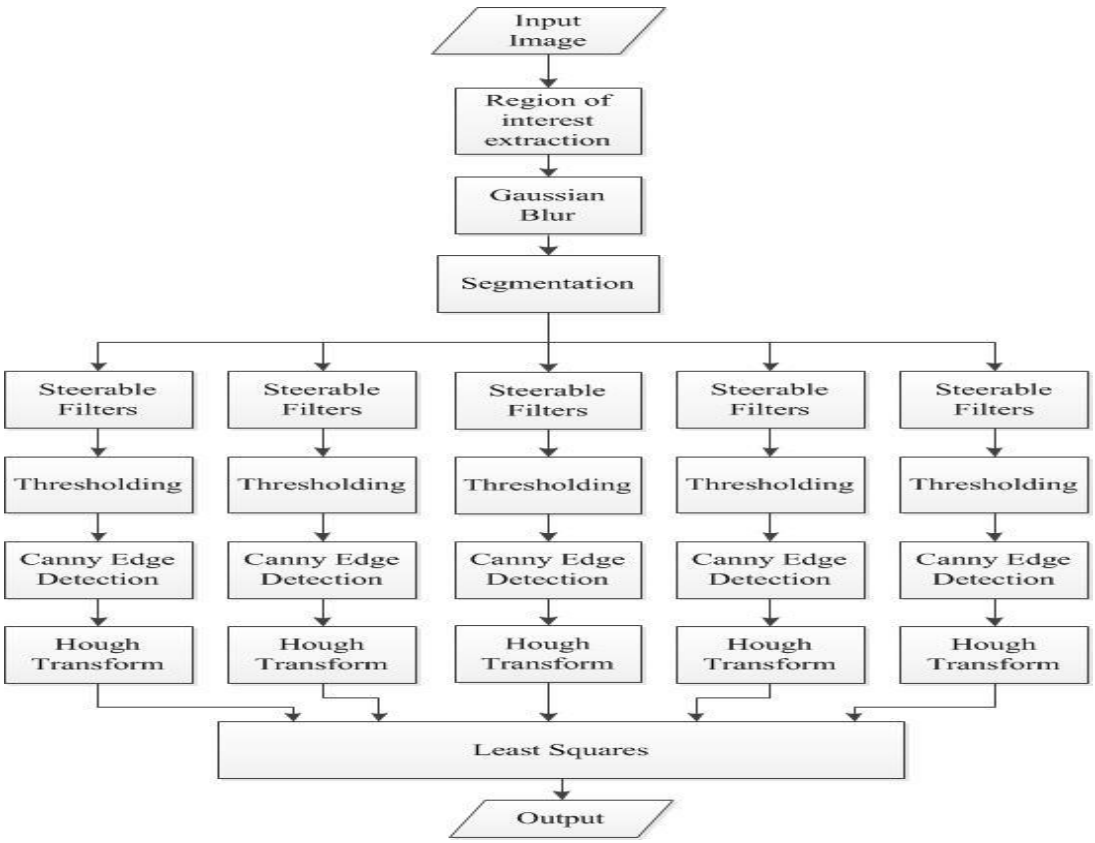

Figure 2.1 : Flow chart of the image processing module.

2.1 Region of Interest (ROI) Extraction and Segmentation

Extracting a reliable ROI from the given image dramatically increases the performance by removing unnecessary and wrong features from the road area in the image during the feature extraction in addition it causes the reduction of the computational cost. The size of ROI is determined by the covariance values of the lateral position of the control points obtained from extended Kalman Filter (EKF). Control points and the covariance values that are used in ROI calculation process are explained in detail in the section 3.2. The mask that is used to define the ROI on the image is formed via applying a two-dimensional Gaussian distribution on the control points. Two-dimensional Gaussian's mean is calculated and assigned to control points and the covariance is the values obtained from EKF. This type mask is illustrated in Figure 2.2. In order to calculate the ROI, the mask and the input image is multiplied element wise. The resulting ROI after multiplication is shown in Figure 2.3.

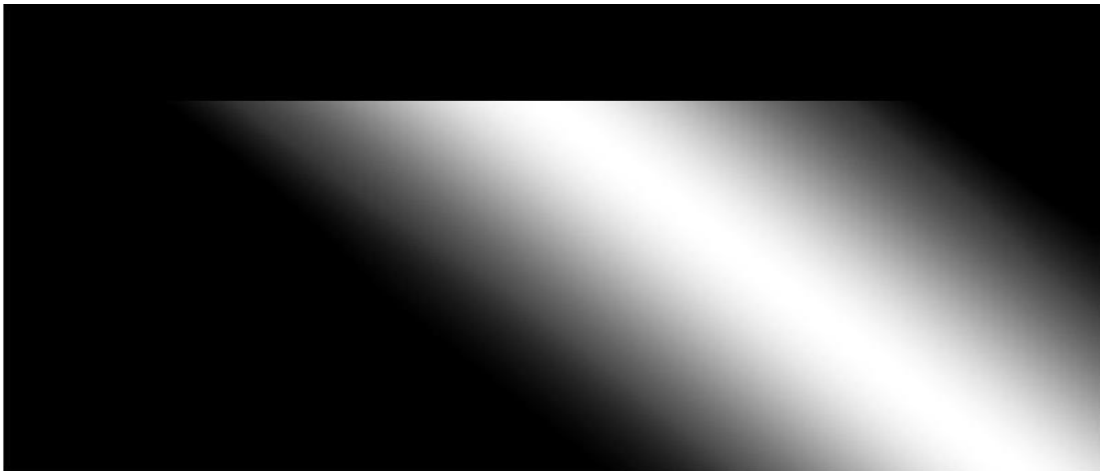

Figure 2.2 : Image of the formed Gaussian mask.

Figure 2.3 : Resulting ROI after the masking operation.

By using the covariance values obtained from EKF, the ROI's size is changed according to the measurements. Covariance values increase when no measurements are obtained which leads to a larger ROI to be calculated. On the other hand, as measurements are obtained consecutively, the covariance becomes smaller and the ROI starts contracting, reducing computational cost.

Because, the steerable filters used in feature extraction part is applicable to lanes with constant width. The input image is divided into five segments horizontally assuming the lane width is constant in each segment. Due to the perspective on the image, the lane's width gets smaller as the lanes get further away in the field of view. There is two ways to resolve this perspective transform. First is converting the input image to a bird eye view using inverse perspective mapping (IPM). However, this method implies that one should know the exact position of the camera according to the world frame. Because, the system is aimed to be a module that can be mounted on different vehicles it is assumed that there will be no such information available for the algorithm. Therefore, the image is divided into five segments where the lanes have relatively constant width. Lowest point on the image plane is the closest to the vehicle given, as the segments go up on the image plane the size of the segments decreases. This is because the rate of change on the lane width increases as segments refer to places that are more distant from the vehicle due to the perspective effect. The segments on the input image is demonstrated in Figure 2.4.

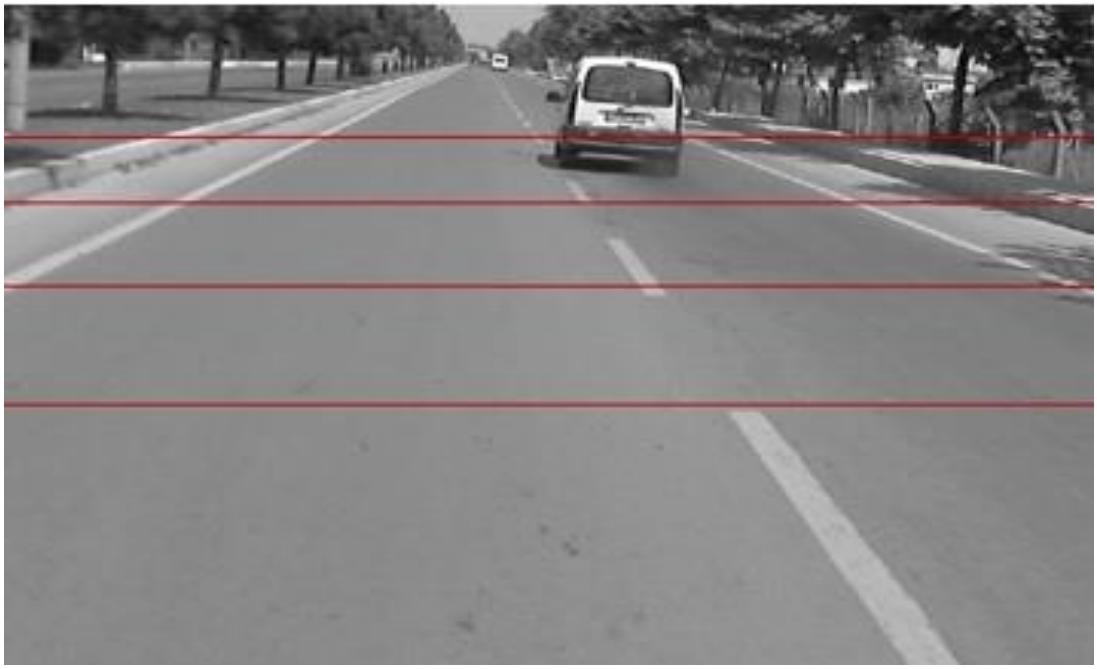

Figure 2.4 : Segmentation of the input image.

2.2 Lane Feature Extraction

After the segmentation and the ROI is formed and the lane features are extracted. Lane features are extracted in two steps. First, steerable filters are applied to the image and then Canny Edge detection is performed. Because of its computational speed and robustness in filtering performance, steerable filters are chosen among the many alternative methods for filtering and the Canny edge detection is performed to suppress the found edges to one pixel width. This property increases the Hough transform performance because it reduces the number of peaks in the Hough space.

In image processing tasks, such as texture analysis, edge detection, motion analysis and image enhancement, oriented filters are widely used. Using oriented filters for texture analysis enables the algorithm to extract features in particular orientations. Adaptively updating the orientation of the filter the algorithm becomes able to track the lanes. Steerable filters are defined as a class of filters that any of arbitrary orientation can be constructed as a linear combination of a set of "basis filters" [33]. As indicated in [33], a function must satisfy Equation 2.1 in order to be steerable.

$$f^{\theta}(x,y) = \sum_{j=1}^{M} k_j(\theta) f^{\theta_j}(x,y) \qquad \textbf{(2.1)}$$

Proving if a function $f(x,y)$ is steerable, finding which number must M assume is required to define the rotation of the function. It also needs to be defined what the interpolation functions $k_j(\theta)$ must be. These steerability requirements are explained in detail in [33]. The Steerable filters that are used in this thesis for lane feature extraction is constructed from the second derivatives of 2D Gaussians. However, steerable filters can be constructed from arbitrary functions, which should satisfy 2.1.

Steerable filters that are constructed via second derivatives of the 2D Gaussians have numerous aspects that makes this filter desirable for lane tracking applications. First, as mentioned previously steerable filters are constructed by a linear sum of a number of basis filters multiplied by interpolation functions $k_j(\theta)$. Second, three basis filters and some particular interpolation functions can construct the derivative of two-dimensional Gaussians steerable filter $k_j(\theta)$, which is used to define the rotation of the filter. Therefore, the basis filters can be constructed separately in order to speed up the computation in a real time system. In addition to creating basis filters separately,

the output image after filtering via steerable filters can be computed by taking convolution of the input image with basis filters at the same time and calculating the linear sum of the output with interpolation functions. This process is illustrated in the Figure 2.5.

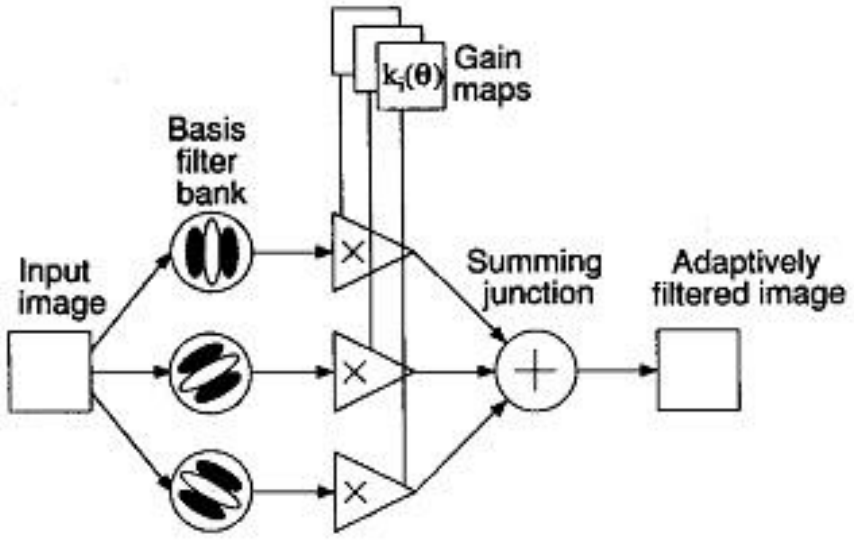

Figure 2.5 : Block diagram of steerable filtering algorithm [33].

In Figure 2.5, the second important aspect of steerable filters is also illustrated. From three basis filters with certain gain maps, the lanes at any orientation on the image can be filtered. The gain maps on Figure 2.5 refers to the orientation values of the steerable filters. The formulation of Steerable filters are given in Equations 2.2, 2.3 and 2.4 [33].

$$G_{xx}(x,y) = \frac{\partial^2}{\partial x^2} e^{\frac{-(x^2+y^2)}{\sigma^2}} = -\left(\frac{2x^2}{\sigma^2} - 1\right)\frac{2}{\sigma^2} e^{\frac{-(x^2+y^2)}{\sigma^2}} \quad \textbf{(2.2)}$$

$$G_{xy}(x,y) = \frac{\partial}{\partial x \partial y} e^{\frac{-(x^2+y^2)}{\sigma^2}} = \frac{4xy}{\sigma^4} e^{\frac{-(x^2+y^2)}{\sigma^2}} \quad \textbf{(2.3)}$$

$$G_{yy}(x,y) = \frac{\partial^2}{\partial y^2} e^{\frac{-(x^2+y^2)}{\sigma^2}} = -\left(\frac{2y^2}{\sigma^2} - 1\right)\frac{2}{\sigma^2} e^{\frac{-(x^2+y^2)}{\sigma^2}} \quad \textbf{(2.4)}$$

Equation 2.2 is the partial derivative of the two dimensional Gaussian with respect to x and Equation 2.3 features the partial derivative of the two dimensional Gaussian with respect to x and y. Lastly Equation 2.4 is the partial derivative of the two dimensional Gaussian with respect to y. In Figure 2.6 the image intensity representations of

Equations 2.2, 2.3 and 2.4, which are the basis set for the steerable filters, are illustrated.

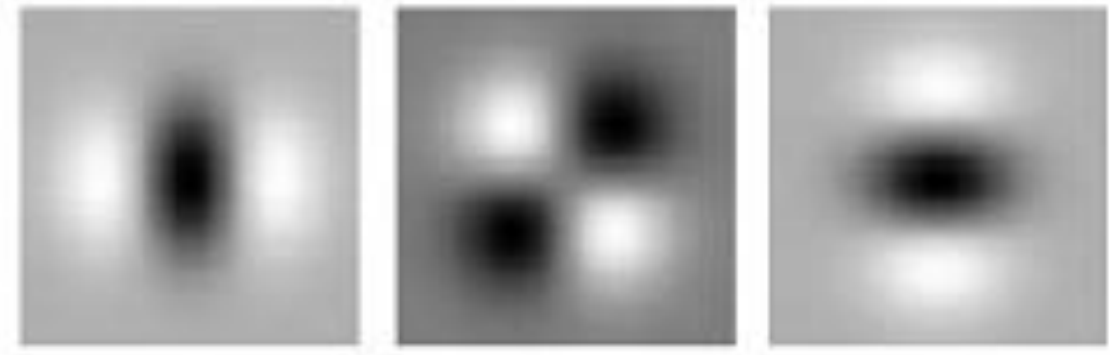

Figure 2.6 : Result of equations 2.1, 2.2 and 2.3 from left to right.

In [33], it has been shown that the response in any rotation of the filter can be calculated using Equations 2.2, 2.3 and 2.4. Equation 2.5 defines the filter response in any rotation θ.

$$G2^{\theta}(x,y) = G_{xx}\cos^2(\theta) + G_{yy}\sin^2(\theta) + G_{xy}\cos(\theta)\sin(\theta) \qquad \textbf{(2.5)}$$

Application of steerable filters on the input image requires two controllable parameters: (i) rotation of the filter θ and (ii) the kernel size of the Gaussian σ. As explained previously, the steerable filters are by nature are tuned to find lane features with a certain width. This property depends on the variance σ. For each segment created in the previous step, a particular steerable filter according to the expected average width of the lanes is used by defining a certain σ for each filter. For detecting the lanes, we use the knowledge that the filter response in the direction of the lane should be near maximum and the filter response in the vertical direction should be relatively much lower. While applying steerable filters, directions of the estimated lanes are obtained from EKF algorithm as explained in section 3. Steerable filter is applied in two directions one: (i) is in the estimated lane direction and the other is (ii) vertical to the estimated lane direction. The difference between two filtered images should be high where lanes in the estimated direction is located on the image. Therefore, by thresholding the difference image, the lane features are obtained. Lastly, the estimated feature points regarding to the lanes respective intensity values are checked on the grey scale input image. Knowing that lanes are always painted in high intensity paints such as white or yellow, secondary threshold is applied, in this manner; under a certain intensity valued feature pixels are suppressed to zero in black and white

image. This last thresholding removes the false features detected due to their gradient intensity values. Figures, 2.7, 2.8, 2.9 and 2.10 illustrate the input image, steerable filter output, the thresholded image and lastly, the resulting image after intensity value check respectively.

Figure 2.7 : Input image obtained from webcam.

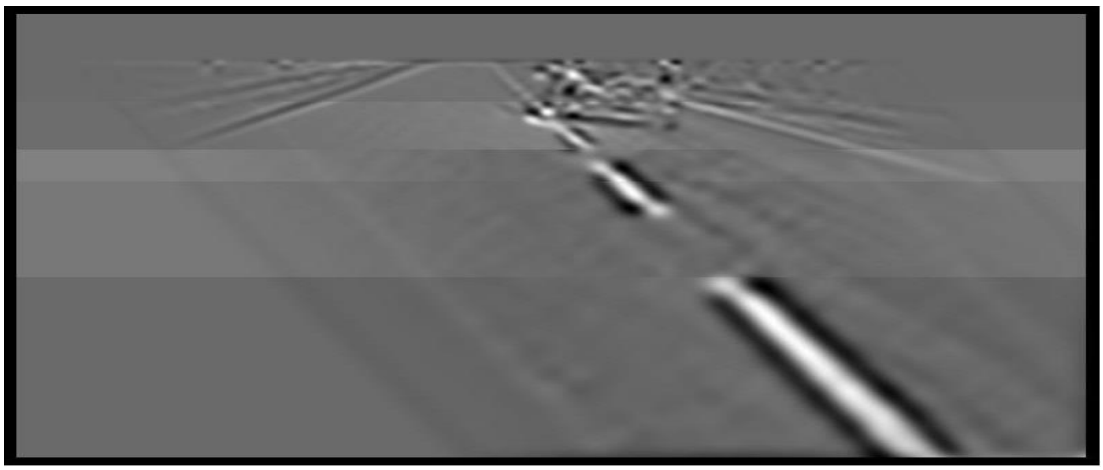

Figure 2.8 : Steerable filter output.

Figure 2.9 : Filter output after thresholding.

Figure 2.10 : Binary image after the intensity value check, note that some false positives are removed.

The steerable filter output is normalized during the process for thresholding. However, if there is no line in the orientation of the filter is apparent the filter output yields to an image containing close to minimum values. Normalizing this filter output result in an image containing intensity values close to the maximum even though no line is apparent. In these cases, checking the intensity values have greater effects on the performance of the system. This phenomenon is shown in the figures 2.11 and 2.12, the binary images before intensity check and after intensity check respectively.

Figure 2.11 : Binary image before the intensity check.

Figure 2.12 : Binary image after the intensity check.

In order to obtain more robust results from Hough transform due to previously explained reasons Canny edge detection is applied to the filtered image. Canny Edge detector is a very common method for finding edges on an image, it is first proposed in 1983 by Canny [34]. Since it is a very common method, there are packages in MATLAB or OpenCV that apply Canny Edge Detection on images. Every one of these packages apply the same steps with some variations that will be explained in the following paragraphs.

Canny edge detection is a multi-stage algorithm that utilizes the edge gradients and gradient directions calculated via Sobel Operator [35]. Initially a Gaussian smoothing is applied to reduce noise on the image before calculating the edge gradients. The edge strength is calculated by taking the gradient of the image via Sobel Operator. Two different 3x3 masks are used in the Sobel Operator that calculates the gradient in the x-direction (columns) and in y-direction (rows). These masks are given in Equations 2.6 and 2.7.

$$G_{sobel_x} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 2 \end{bmatrix} \quad \textbf{(2.6)}$$

$$G_{sobel_y} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & 2 & -1 \end{bmatrix} \quad \textbf{(2.7)}$$

Using Equations 2.6 and 2.7, the edge strength or in other words the gradient of the image is calculated using the Equation 2.8.

$$\left|S_{gradient}\right| = \left|G_{sobel_x}\right| + \left|G_{sobel_y}\right| \quad \textbf{(2.8)}$$

The third step is to calculate the edge direction, which is calculated by taking the inverse of the tangent of the calculated gradients as shown in Equation 2.9. Equation 2.9 will give an error if both gradients are 0, therefore a hard constraint is made. While G_{sobel_x} is equal to zero, if G_{sobel_y} is zero the gradient is set to 0 degrees otherwise it is set to 90 degrees in order to neutralize this error.

$$\theta_{Sobel} = \tan^{-1}\left(\frac{G_{sobel_y}}{G_{sobel_x}}\right) \quad \textbf{(2.9)}$$

After the gradient direction is calculated, the direction of the gradient is rounded to a value that can be traced on the image. Since images are pixel arrays, there are four possible directions that the gradient direction can be rounded to some specific values as illustrated in the Figure 2.13.

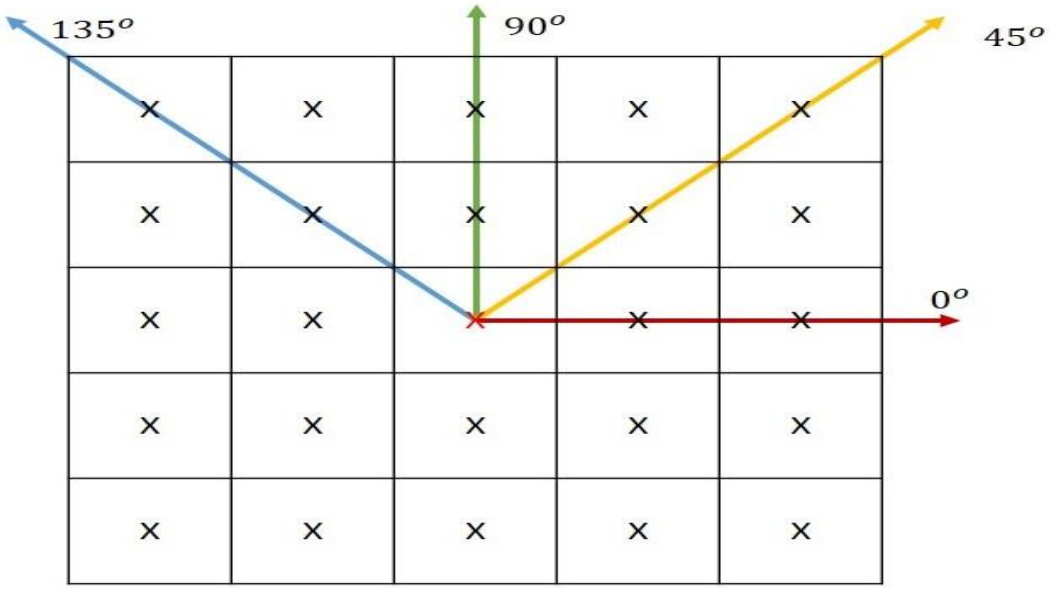

Figure 2.13 : Four trackable directions on the image.

The gradients of each pixel are rounded as follows, edge directions between 0 to 22.5 and 157.5 to 180 degrees is set to 0 degrees, the red direction on Figure 2.13. Edge directions between 22.5 to 67.5 is set to 45 degrees, the yellow direction on Figure 2.13. Edge directions between 67.5 to 112.5 degrees is set to 90 degrees , the green direction on Figure 2.13 and finally edge directions between 112.5 to 157.5 is set to 135 degrees which is the blue direction on Figure 2.13.

After the edge directions are rounded to trackable directions on the image, non-maximum suppression algorithm is applied. The gradient directions in trackable directions on image and the magnitudes, which represent the edge strength, are given, the non-maximum suppression algorithm searches for the maximum in the gradient direction of the pixel. If the gradient magnitude of the center pixel is not greater than the two neighbors in the gradient direction it is suppressed to zero. For example in Figure 2.14, the arrow indicates the gradient directions, red value the center pixel has a gradient magnitude of 5 and its two neighbors gradient magnitude along the gradient direction are 25 and 35. The reason for that can be explained considering that the center pixel does not have a gradient magnitude greater than the two neighbors along the gradient direction it is suppressed to 0.

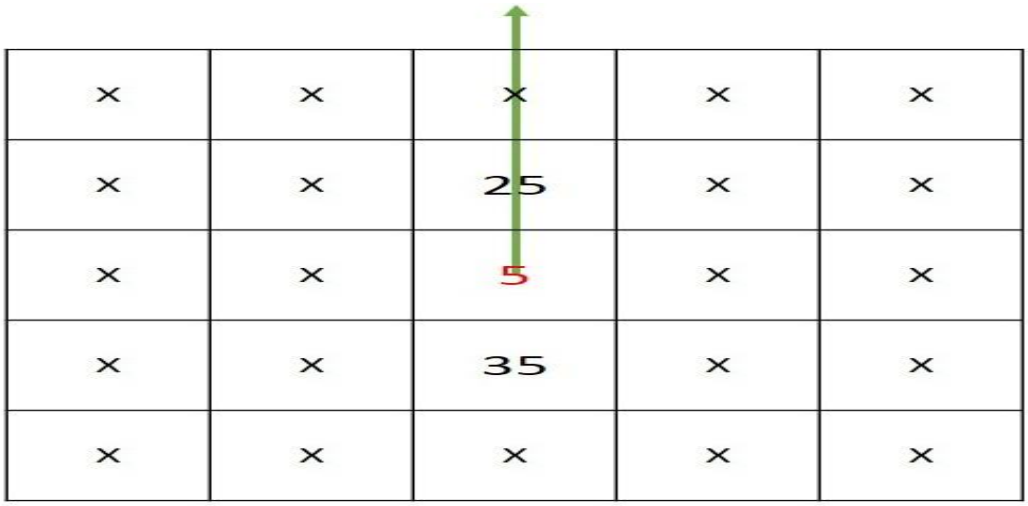

Figure 2.14 : Gradient values and the gradient direction of the center pixel.

Edge contours can sometimes be broken, creating dashed lines on the edges because of the operator output fluctuating above and below the threshold, this phenomenon is called streaking. Streaking is eliminated by applying two different thresholds on the image defined as T_1 and T_2. T_1 is the threshold with a greater value and T_2 is the threshold with a lower smaller value. In order to assume a pixel is an edge as a starting point, the pixel must have a gradient magnitude higher than T_2, than any pixel that is connected to the starting pixel is assumed to be an edge until a pixel that has a gradient magnitude lower or equal to the low threshold T_2 is come across in the search algorithm. An example result of Canny Edge Detection after the steerable filter step is illustrated in Figure 2.15.

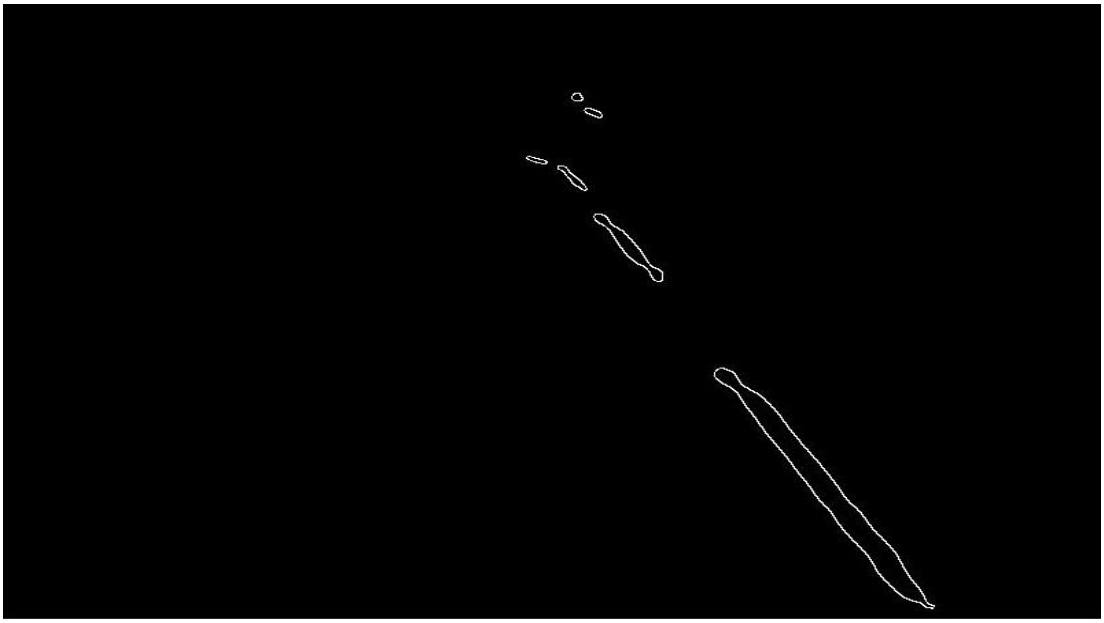

Figure 2.15 : Output of the Canny Edge Detection algorithm.

2.3 Outlier Removal and Parameter Estimation

Outlier removal and model parameter estimation is the final stage of the detection algorithm. The estimated lane parameters are fed to the EKF system as measurements. Hough Transform is used for the outlier removal process it is applied separately on each segment on the image. This produces fractal lines on the whole image. After applying Hough Transform, the parameters of the lane model are estimated via Least Squares algorithm performed on the found fractal lines.

Hough Transform was originally developed to make an analysis of bubble chamber photographs in 1959, which is initially developed to find lines on an image [36]. A line equation as in Equation 2.10, has parameters x and, and constants slope m and zero crossing point c.

$$y = mx + c \tag{2.10}$$

It is known that from a single point infinite numbers of lines pass as illustrated in Figure 2.16.

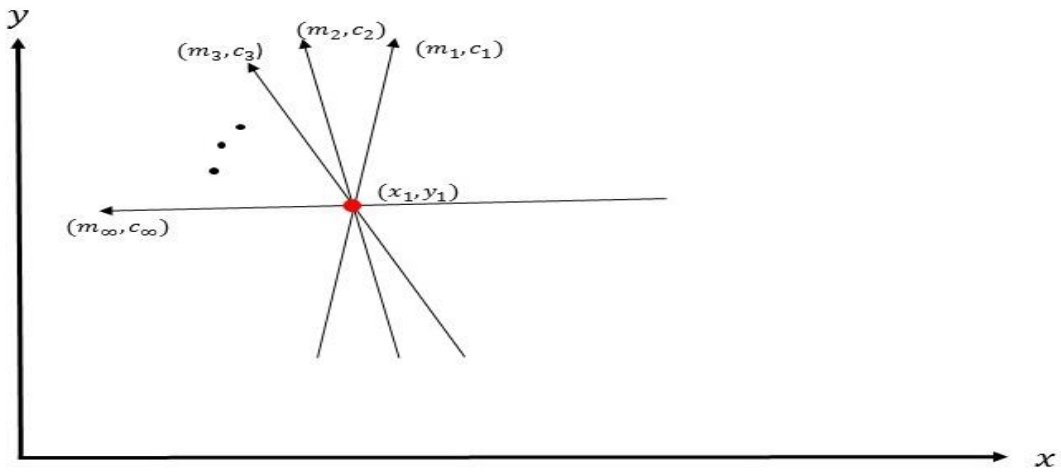

Figure 2.16 : Passing lines from a single point.

The Hough Transform changes the parameter space and defines the line in terms of slope (m) and zero crossing (c) point instead of x and y. If a simple modification is made on Equation 2.10, the line Equation 2.11 in (m, c) domain is obtained for the point (x_1, y_1).

$$c = y_1 - mx_1 \tag{2.11}$$

The graph of the Equation 2.11 is shown in Figure 2.17. It can be seen that a single point that has infinite number of passing lines in (x, y) domain refer to a single line in (m, c) domain. Using this property of Hough transform, any lines with the same slope and offset (i.e. orientation) values could be identified on the image.

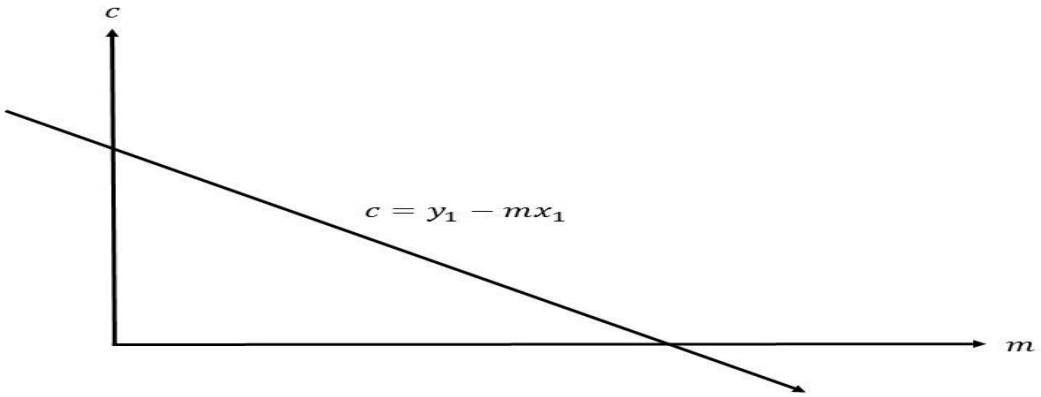

Figure 2.17 : Point (x_1, y_1) in (m, c) domain.

In order to find lines on the image, Hough Transform needs the input image to be a binary image. After every pixel is transformed in the (m, c) domain as illustrated in Figure 2.18, a histogram is made for the crossing points for of the lines in (m, c) domain. Histogram bins that have more votes than a previously defined threshold correspond to the strong lines on the image. However, the initially proposed method fails when there is vertical lines on the image. Vertical lines makes the method unable to transform the line to the (m, c) domain because of the infinite m value for a vertical line.

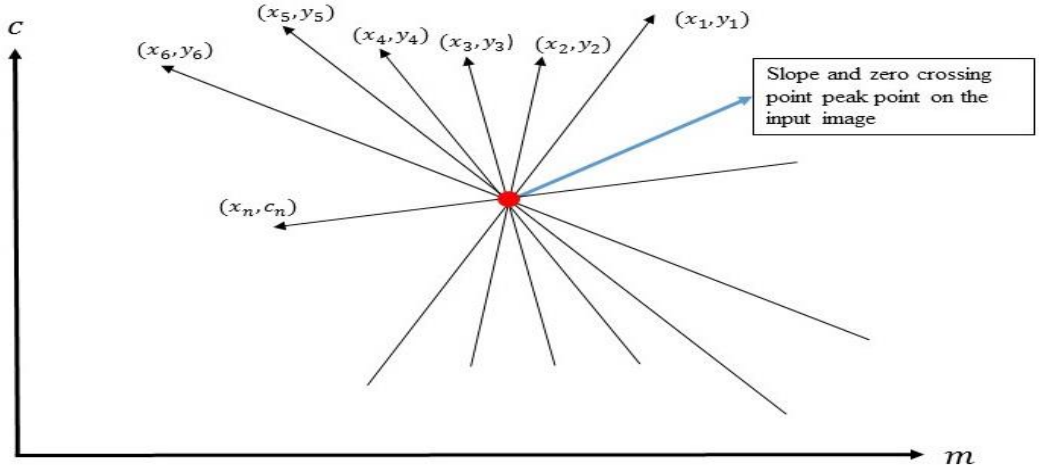

Figure 2.18 : 'n' number of points represented in (m, c) domain.

In order to solve this problem caused by vertical lines on the image, the line is defined on the polar coordinates rather than the cartesian coordinates. Polar coordinates defines

a point with the distance from the origin r and the angle of the direction of the distance ϕ. Transformation between cartesian coordinates and the polar coordinates are given in Equations 2.12 and 2.13.

$$x = rcos(\phi) \quad \textbf{(2.12)}$$

$$y = rsin(\phi) \quad \textbf{(2.13)}$$

A line on polar coordinates are defined by the distance r between the closest point of the line to the origin and the angle ϕ which is the angle of the orthogonal vector from origin to the line pointing towards upper half plane. ϕ is the angle of the vector pointing to the closest point of the line to the origin if the line is located in the positive half of the coordinates. By using normal parameterization, described thoroughly in [37] the line equation can be written as seen in Equation 2.14.

$$r = xcos(\phi) + ysin(\phi) \quad \textbf{(2.14)}$$

Using this parameterization in Equation 2.14, one can project the image from cartesian coordinate system to polar coordinates (r, ϕ) system, which now contain harmonics instead of line in the projected space. This also solves the vertical line problem on the image. An example result of Hough Transform in polar coordinates are illustrated in the Figure 2.19.

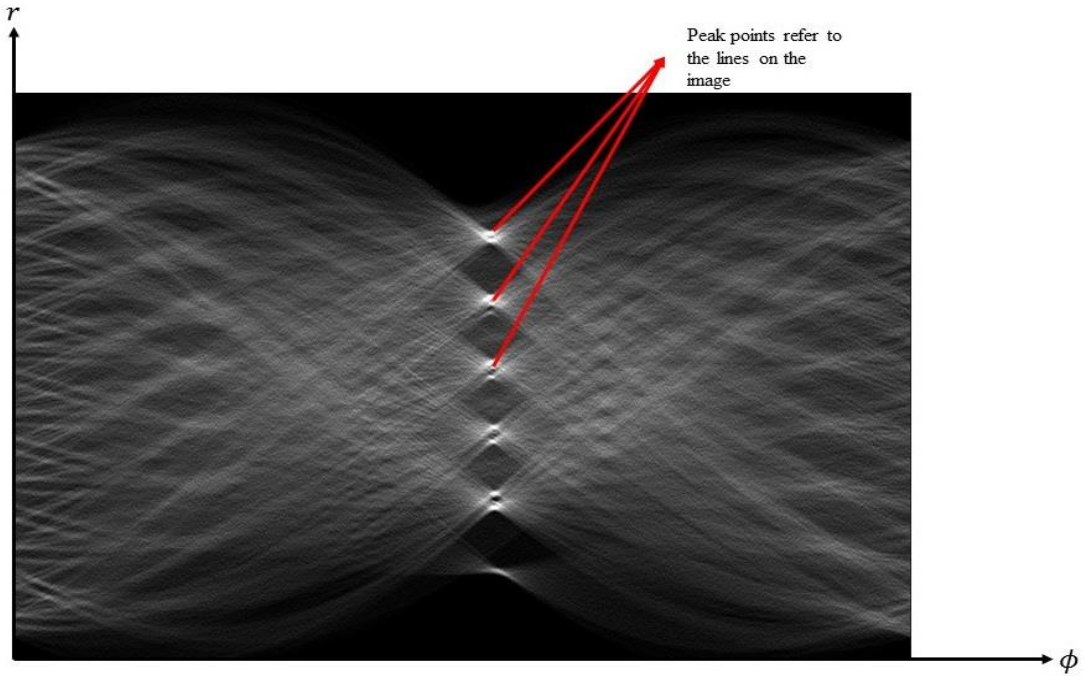

Figure 2.19 : Hough Space representation of an image.

In Hough Space, the horizontal axis ϕ is between -90 and 90 if no restriction is made. However, ϕ can be restricted between particular values in order to find lines in certain angles on the image. This restriction is made regarding to some previously known geometric properties of lanes on the image plane. According to this, expected lane positions on the image plane are illustrated in Figure 2.20.

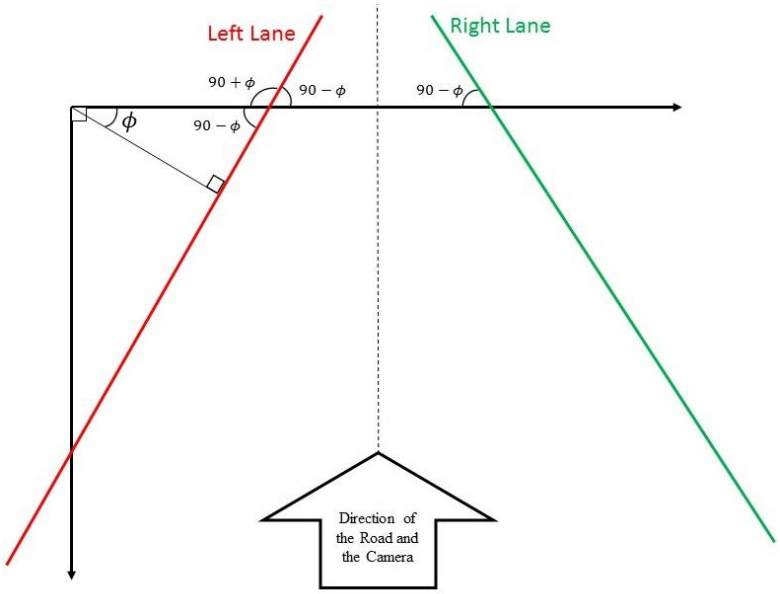

Figure 2.20 : Hough Transform angle ϕ for left and right lane.

By assuming that lanes cannot be horizontal on the image plane while the camera is looking in the direction of the road and left and right lanes are approximately symmetrical to each other according to a virtual vertical line indicated by dashed lines on Figure 2.20, the constraint on ϕ is defined. Regarding to these assumptions ϕ is fixed between -70 degrees and 70 degrees. With this constraint, Hough Transform searches for the peaks in Hough Space only between -70 degrees and 70 degrees.

Fractal lines regarding to the lanes for each segment is found via Hough transform, however a model is need to fit into this fractal lines in order to represent the lane on the whole image. The fitting of the detected parameters on the lane model is achieved by using the Least Squares method. The Least Squares is a method that fits a model to a set of measurements by minimizing the squared differences between the measurements and the model prediction.

Linear Least Squares method is used to fit a model to the data available. In order to explain Least Squares method in more detail some basic concepts about statistics need

to be defined. If a sequence of data $x_1, x_2, \dots, x_N$ given, the mean (expected value) is defined as in Equation 2.15.

$$\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x_n \tag{2.15}$$

Mean of a data set x is denoted as $\bar{x}$. Data sets of same mean can have different variations about the mean of the data; in order to denote this the concept of variance is introduced. The variance of a data is defined for data set x in Equation 2.16.

$$\sigma_x^2 = \frac{1}{N}\sum_{n=1}^{N} (x_n - \bar{x})^2 \tag{2.16}$$

In addition to mean and variance, the standard deviation is the square root of the variance as given in Equation 2.17.

$$\sigma_x = \sqrt{\frac{1}{N}\sum_{n=1}^{N} (x_n - \bar{x})^2} \tag{2.17}$$

A model have to be defined for the lanes in order to apply Least Squares method. The lanes are modelled as first order polynomials as given in Equation 2.18.

$$y = ax + b \tag{2.18}$$

When the data set, $\{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ that is obtained from Hough Transform, is given, the squared error can be defined as seen in Equation 2.19.

$$E(a, b) = \sum_{n=1}^{N} (y_n - (ax_n + b))^2 \tag{2.19}$$

In order to find the values a and b, that minimizes the error $E(a, b)$, there is a pre-condition that a and b values should satisfy the Equation 2.20 and 2.21.

$$\frac{\partial E}{\partial a} = 0 \tag{2.20}$$

$$\frac{\partial E}{\partial b} = 0 \tag{2.21}$$

Equations 2.22 and 2.23 are obtained by differentiating $E(a, b)$.

$$\frac{\partial E}{\partial a} = \sum_{n=1}^{N} 2\big(y_n - (ax_n + b)\big).(-x_n) \tag{2.22}$$

$$\frac{\partial E}{\partial b} = \sum_{n=1}^{N} 2(y_n - (ax_n + b)).1 \tag{2.23}$$

If Equations 2.23 and 2.22 are re-defined by using 2.20 and 2.21, Equation 2.24 and 2.25 are obtained.

$$\sum_{n=1}^{N} \big(y_n - (ax_n + b)\big).(-x_n) = 0 \tag{2.24}$$

$$\sum_{n=1}^{N} \big(y_n - (ax_n + b)\big).1 = 0 \tag{2.25}$$

Equations 2.24 and 2.25 may be rewritten as in Equation 2.26 and 2.27.

$$\left(\sum_{n=1}^{N} x_n^2\right) a + \left(\sum_{n=1}^{N} x_n\right) b = \sum_{n=1}^{N} x_n y_n \tag{2.26}$$

$$\left(\sum_{n=1}^{N} x_n\right) a + \left(\sum_{n=1}^{N} 1\right) b = \sum_{n=1}^{N} y_n \tag{2.27}$$

Equations 2.26 and 2.27 can be represented as a matrix equation as in equation 2.28.

$$\begin{pmatrix} \sum_{n=1}^{N} x_n^2 & \sum_{n=1}^{N} x_n \\ \sum_{n=1}^{N} x_n & \sum_{n=1}^{N} 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \sum_{n=1}^{N} x_n y_n \\ b \sum_{n=1}^{N} y_n \end{pmatrix} \tag{2.28}$$

The values of a and b which minimizes the error $E(a, b)$ can be obtained by solving the Equation 2.29. Note that the matrix should be invertible, the proof can be found in [38] for further reading.

$$\binom{a}{b} = \begin{pmatrix} \sum_{n=1}^{N} x_n^2 & \sum_{n=1}^{N} x_n \\ \sum_{n=1}^{N} x_n & \sum_{n=1}^{N} 1 \end{pmatrix}^{-1} \begin{pmatrix} \sum_{n=1}^{N} x_n y_n \\ b \sum_{n=1}^{N} y_n \end{pmatrix} \quad \textbf{(2.29)}$$

The procedure of detecting and locating the lanes on image procedure is completed by obtaining the model parameters a and b. Briefly, process starts by creating a ROI around the expected position of lanes then segmenting the image. Next, steerable filters are applied and the filter response is thresholded. Canny Edge detector is used to find edges on the resulting binary image. After the edges are found on the image, Hough Transform is applied and lines referring to the lanes are found for each segment. Lastly, the Least Squares method is applied in order to fit a first order polynomial to the data obtained from Hough Transform.

3. LANE TRACKING

In order to track the lanes the problem is considered as a localization problem as explained in [39]. In our case, we are only interested in lateral localization of the vehicle on the road according to the initial position. In order to achieve this a version of Kalman Filter, the Extended Kalman Filter (EKF), which is for non-linear systems is utilized to track vehicle and lane position. Tracking is made in two phases, first phase involves only the image processing and second phase concerns the fusion image processing with vehicle angular velocity data obtained via IMU. The fusion with IMU is explained in Section 3.4 along with the motion model and measurement model of the fusion algorithm.

3.1 Kalman Filter and Extended Kalman Filter (EKF)

Kalman filter is first proposed in 1960 as a solution for the linear-quadratic problem by R.E Kalman [40]. The linear-quadratic problem is defined as the problem of estimating the state of a system that is exposed to white noise. This problem can also be stated as estimating state of a linear stochastic system by using measurements that are linear functions of the state. A stochastic system is a system that is defined by the collection random variables. Even though, Kalman Filter is particularly for linear systems, an altered version of Kalman Filter, the Extended Kalman Filter (EKF) can be implemented on non-linear systems.

Knowledge of state space modelling is mandatory in order to understand how the Kalman Filter works and guarantees better measurement results. Representing a higher order system by a set of first order differential equations is called the state space representation of a dynamic system [41]. State equations for a linear stochastic system including process model and measurement model are given in Equations 3.1 and 3.2 respectively.

$$x_{k+1} = A_k x_k + B_k u_k + w_k \quad \textbf{(3.1)}$$

$$z_k = H_k x_k + v_k \quad \textbf{(3.2)}$$

Where, x is the state vector, u is the input or control vector and z is the measurement vector. A is the state transition matrix, B is the control or input matrix and H is the measurement matrix. The random variables w_k and v_k represent the process and measurement noise respectively. They are assumed to be independent random variables with normal probability distributions as given in Equations 3.3 and 3.4.

$$p(w) \sim N(0, Q) \quad \textbf{(3.4)}$$

$$p(v) \sim N(0, R) \quad \textbf{(3.5)}$$

In Equation 3.4, it is defined that the random variable w has a normal distribution with zero mean and a covariance of Q. Similarly, in Equation 3.5, the random variable v has normal distribution with a mean of zero and a covariance of R. Q and R are the matrices such that their size depends on the order of the process and measurement model. Kalman Filter is a recursive filter that has two main stages, time update (prediction) and measurement update (correction). The algorithm consists of two Time Update Equations and three Measurement Update Equations. Time Update Equations are given in Equations 3.6 and 3.7. Measurement update Equations are given in 3.8 3.9 and 3.10.

$$\hat{x}_{k+1}^{-} = A_k \hat{x}_k + B_k u_k \quad \textbf{(3.6)}$$

$$P_{k+1}^{-} = A_k P_k A_k^T + Q_k \quad \textbf{(3.7)}$$

$$K_k = P_k^{-} H_k^T (H_k P_k^{-} H_k^T + R_k)^{-1} \quad \textbf{(3.8)}$$

$$\hat{x}_k = \hat{x}_k^{-} + K_k (z_k - H_k \hat{x}_k^{-}) \quad \textbf{(3.9)}$$

$$P_k = (I - K_k H_k) P_k^{-} \quad \textbf{(3.10)}$$

Equation 3.6 and 3.7 are called the predicted state estimate and predicted covariance estimate respectively. Optimal Kalman gain is calculated in Equation 3.8, the last term, which is, $(H_k P_k^{-} H_k^T + R_k)^{-1}$ is the inverse of the residual covariance matrix S. Diagonal values of this matrix refers to the covariance for the indicated state by that

row. Equation 3.9 is the calculation of the updated state estimates using the measurements. Lastly, updated covariance estimate is calculated for the next cycle in equation 3.11. Figure 3.1 illustrates the Kalman Filter cycle.

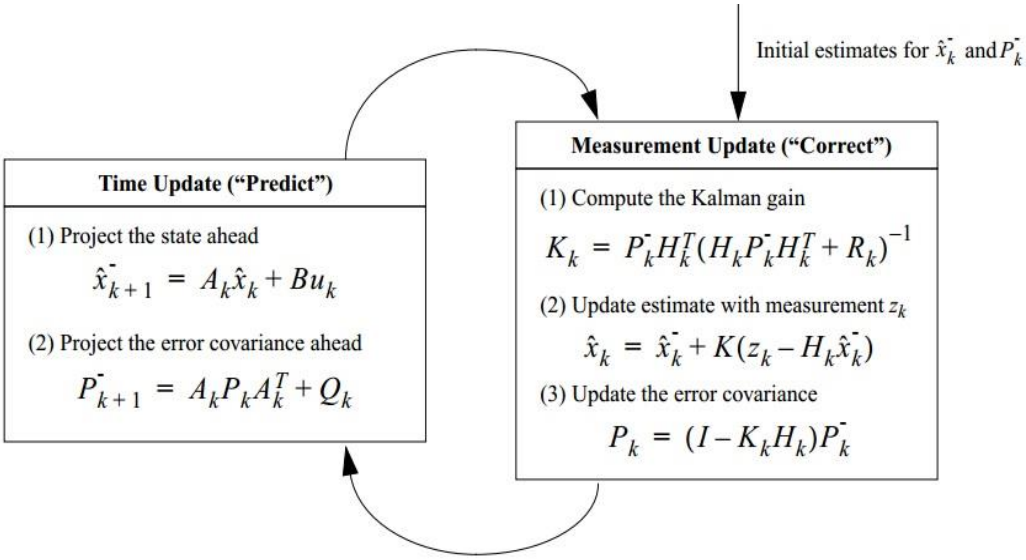

Figure 3.1 : Kalman Filter Cycle obtained from [40].

EKF is the altered version of the Kalman Filter aiming to estimate states in a non-linear system. Equations for EKF algorithm are given in the following equations [40].

The Predicted State Estimate:

$$\hat{x}_{k+1} = f(\hat{x}_k, u_k) \tag{3.11}$$

Predicted Covariance Estimate:

$$P_{k+1}^- = P_{k+1}^- = F_kP_kF_k^T + Q_k \tag{3.12}$$

Near-Optimal Kalman Gain:

$$K_k = P_k^-H_k^T(H_kP_k^-H_k^T + R_k)^{-1} \tag{3.13}$$

In Equations, 3.12 and 3.13 the terms F and H are the Jacobians of the process and measurement functions $f(x,u)$ and $h(x)$, given in the equations 3.14 and 3.15 respectively.

$$F = \frac{\partial f}{\partial x} \tag{3.14}$$

$$H = \frac{\partial h}{\partial x} \quad \textbf{(3.15)}$$

Updated State Estimate:

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-)) \quad \textbf{(3.16)}$$

Updated Covariance Estimate:

$$P_k = (I - K_k H_k)P_k^- \quad \textbf{(3.17)}$$

EKF cycle is almost same as the Kalman filter cycle. Only difference is the EKF equations are inserted into the cycle for calculating predicted state estimate and measurement residual along with calculating the Jacobians of state transition and measurement matrices.

Motion and measurement models for EKF algorithm is explained in sections 3.2 and 3.3 respectively. In the motion model, section the constant velocity and position models are explained. In the measurement model section the pinhole camera model and the projection between the image frame and the world plane are explained. Image plane is illustrated in Figure 3.2 where u refers to the columns on the image and v refers to the rows on the image. The world plane is illustrated in Figure 3.3 where, z_w refers to the axis in the direciton of the road and x_w indicates horizontal axis.

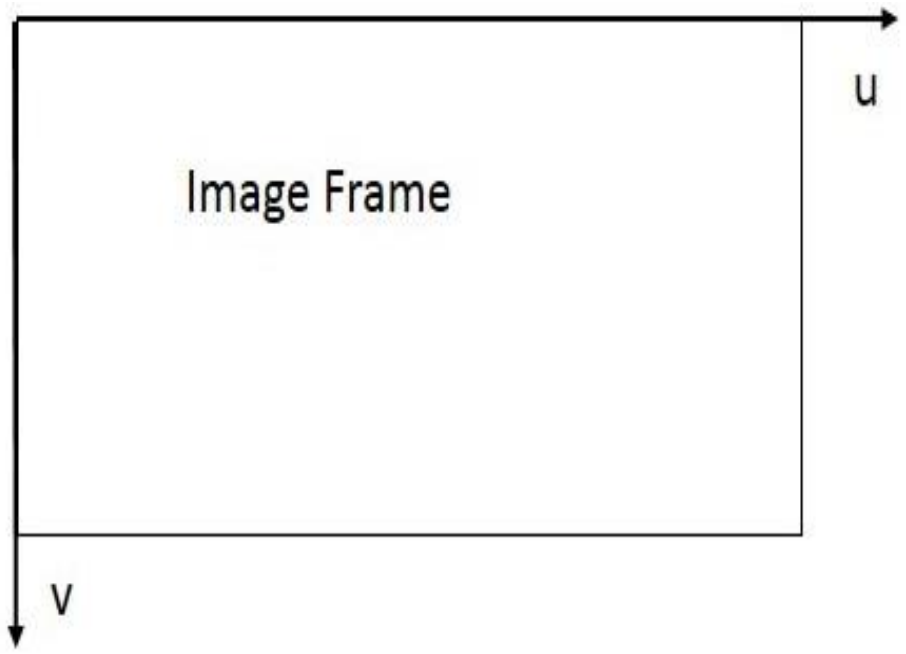

Figure 3.2 : Image frame axes.

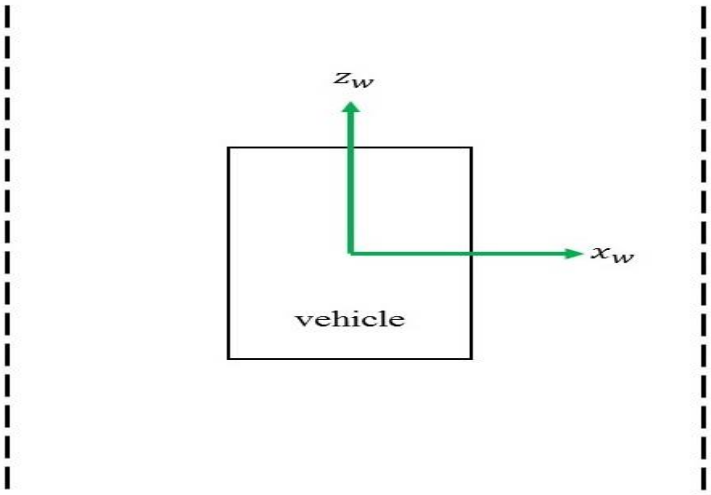

Figure 3.3 : World plane axes.

3.2 Motion Model for EKF Algorithm on Image Stream-Only Case

The motion model of the EKF algorithm can be divided into smaller parts. These parts involve the vehicle's motion model and the lanes motion model. In order to model the vehicle, the constant velocity model is preferred. Constant velocity approach does not necessarily mean that the motion is always on constant velocity [39]. It is assumed that the unknown accelerations occur in a Gaussian profile. In other words, accelerations are considered as noise signal in EKF algorithm. While modelling the vehicle motion it is assumed that the vehicle and the camera moves identically, meaning there is no relative movement between the camera and the vehicle. Additionally, for the sake of simplicity and considering that our data is on high-speed, it is assumed that the camera has a constant yaw, pitch and roll with respect to the world plane. The yaw, pitch and roll between the world plane and camera frame is calibrated on a known test frame. On high-speed ignoring camera yaw according to world plane might not cause distortions on the model however, on low speeds lane changing can induce significant yaw in the camera motion. This is because, as the vehicle gets slower more steering angle is to be induced in order to move laterally which results in inducing yaw to the system. States regarding to the motion of the vehicle is given in Equation 3.18 where, x denotes the lateral position of the vehicle on the road and $\dot{x}$ denotes the lateral velocity of the vehicle.

$$x_{vehicle} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \tag{3.18}$$

Motion the lanes are modelled by the motion of representative control points referring to the lanes. These control points are sampled from the extracted lanes by the image processing algorithm. Five control points are sampled for each lane, which are at pre-determined distances of 5m, 10m, 15m, 20m and 25m ahead of the vehicle on the world plane. Number of control points are chosen considering the real time computation limitations. State vector regarding to the control points sampled from the lanes are given in Equation 3.19.

$$x_{control\ points} = \begin{bmatrix} P_{1_x} \\ \vdots \\ \vdots \\ P_{5n_x} \end{bmatrix} \tag{3.19}$$

In Equation 3.19, P_{m_x} denotes the m^{th} control point in the P's x axis position according to the world plane and n refers to the number of tracked lanes. For example, if the algorithms is tracking two lanes the vector in Equation 3.19 will have ten elements. If states in Equations 3.18 and 3.19 is combined the state vector of the whole motion model is obtained which is given in Equation 3.20.

$$\underline{x} = \begin{bmatrix} x \\ \dot{x} \\ P_{1_x} \\ \vdots \\ \vdots \\ \vdots \\ P_{5n_x} \end{bmatrix} \tag{3.20}$$

The constant velocity model of the vehicle's state update functions are given in Equations 3.21 and 3.22:

$$f_1 = x_{k+1} = x_k + (\dot{x_k} + \alpha)\Delta t \tag{3.21}$$

$$f_2 = \dot{x}_{k+1} = \dot{x}_k + \alpha \tag{3.22}$$

The term α in Equations 3.21 and 3.22 refers to the lateral acceleration, which provides the introduction of lateral acceleration in to the constant velocity model as noise. Step time Δt is chosen as $1/\text{fps}$ of the image stream in the experimental data. State update functions of the control points are defined according to the constant position model. Assuming that the lanes have constant position on world plane is practically true.

Moreover, even though the constant position model is applied to the control points, all the control points are updated according to the measurements in EKF independent of the initial values. The state update function of control points are given in Equation 3.23.

$$f_{i+2} = P_{i_{x_{k+1}}} = P_{i_{x_k}} \qquad i = 1,2 \dots ,5n \tag{3.23}$$

State update functions in Equations 3.21, 3.22 and 3.23 are linear. However, measurement model is a non-linear model therefore; EKF is applied to track the lanes and vehicle position. The state transition matrix is obtained by calculating the Jacobian of the state update functions according to the state vector, which is given in Equation 3.24. The resulting matrix of the calculation in Equation 3.24 is given in Equation 3.25.

$$F = \begin{bmatrix} \frac{\delta f_1}{\delta x} & \frac{\delta f_1}{\delta \dot{x}} & \frac{\delta f_1}{\delta P_{1_x}} & \cdots & \frac{\delta f_1}{\delta P_{nx5_x}} \\ \frac{\delta f_2}{\delta x} & \frac{\delta f_2}{\delta \dot{x}} & \frac{\delta f_2}{\delta P_{1_x}} & \cdots & \frac{\delta f_2}{\delta P_{nx5_x}} \\ & \vdots & & \ddots & \vdots \\ \frac{\delta f_{(nx5)+2}}{\delta x} & \frac{\delta f_{(nx5)+2}}{\delta \dot{x}} & \frac{\delta f_{(nx5)+2}}{\delta P_{1_x}} & \cdots & \frac{\delta f_{(nx5)+2}}{\delta P_{nx5_x}} \end{bmatrix} \tag{3.24}$$

$$F = \begin{bmatrix} 1 & \Delta t & 0 & \dots & 0 \\ 0 & \ddots & & & \vdots \\ \vdots & & \ddots & & \vdots \\ \vdots & & & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}_{(5n+2)x(5n+2)} \tag{3.25}$$

From EKF equations, we know that EKF uses Q in order to calculate the predicted covariance estimate P^-. Predicted covariance estimate is used to calculate the Kalman Gain K, which is used to update the states. In the definition of process noise covariance matrix Q, previously mentioned assumption on the existence of undetermined lateral accelerations in a Gaussian profile is introduced. The calculation in order to obtain Q is given in Equation 3.26.

$$Q = GP_nG^T \tag{3.26}$$

Covariance of the noise vector P_n defines the rate of growth of uncertainty in the model. As P_n increases, the uncertainty in the system increases meaning that the camera is exposed to higher accelerations. Setting P_n small is suitable while travelling inside a lane; however, setting a very small P_n may result in failures during lane changes. For setting P_n, an empirical approach is taken after examining the experimental data of more than 10hrs of highway driving containing at least 250 lane-change events. In Equation 3.26, the matrix G is calculated by the Jacobian of the state update equations according to the noise vector. In our case, the noise vector only consist of the lateral noise on velocity α. Therefore, the Jacobian calculation is given in Equation 3.27 yields to a column vector.

$$G = \begin{bmatrix} \frac{\partial f_1}{\partial \alpha} \\ \frac{\partial f_2}{\partial \alpha} \\ \vdots \\ \vdots \\ \frac{\partial f_{(nx5)+2}}{\partial \alpha} \end{bmatrix} \quad \textbf{(3.27)}$$

The result of the calculation in Equation 3.27 is given in Equation 3.28.

$$G = \begin{bmatrix} \Delta t \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{(5n+2)x1} \quad \textbf{(3.28)}$$

3.3 Measurement Model

The control points mentioned in previous section are regarded as the measurements of the system. The control points found by the image processing module are defined on the image plane however; the measurements have to be in the same plane as the system mode. Since the system model is defined with respect to the world plane, the control points are required to be projected from image plane to world plane. This is achieved in two steps, first projection from image plane to camera frame is performed then, projection from camera frame to world plane is calculated. In order to achieve the projection form image frame to camera frame, pinhole camera model is used and the

projection between the camera frame and the world frame is made by rotation and translation calculations.

Pinhole camera model is the simplest and most common camera model [42]. In this model, a point Q on the camera frame O_c is projected on to an image frame as illustrated in Figure 3.4.

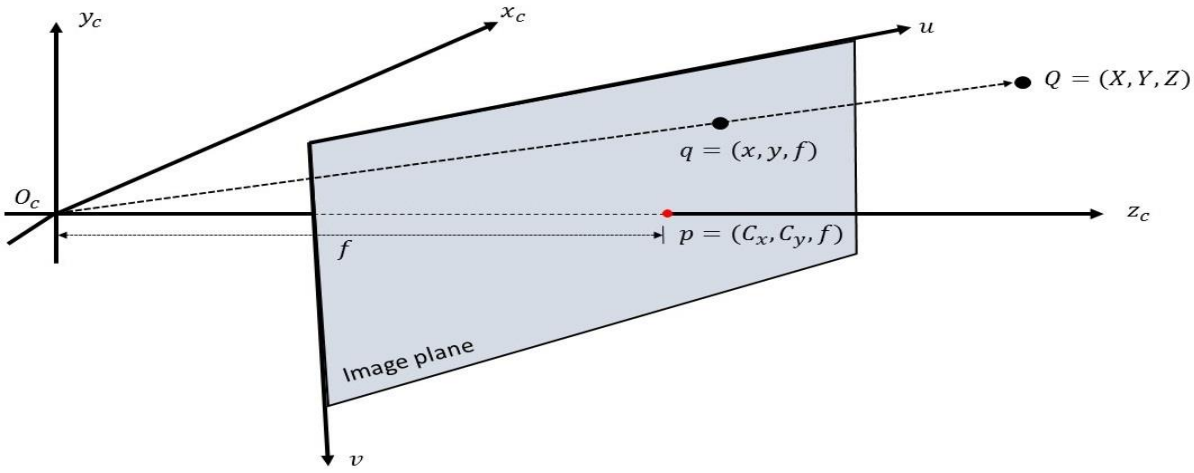

Figure 3.4 : Geometric model of a pinhole camera obtained from [42].

In Figure 3.4, camera frame origin is defined as O_c. The z axis of the camera frame is known as the principal axis. The point where the principal axis intersects the image plane is known as principal point and denoted as p. The translation between the origin of the camera frame O_c and the principal point p is called the focal length and denoted as f. The principal point and focal length are called the intrinsic parameters and can be found via camera calibration. Principle point p is not always located at the center of the image plane because the center of the image sensor is not usually on the camera frame origin O_c. Therefore it has a u and v axis coordinates on image plane denoted by C_x and C_y respectively. The focal length is denoted as only one value on the Figure 3.4, however focal length has two different. This is because , the the imaging elements are often rectangular and the focal length is calculated as the product of the physical focal length F and the size of the corresping edge every imager element s_x and s_y. The equations regarding to focal lengths are given in Equations 3.29 and 3.30.

$$f_x = F s_x \tag{3.29}$$

$$f_y = F s_y \tag{3.30}$$

It should be noted that s_x and s_y cannot be measured individually by any calibration method. Only the f_x and f_y can be derived via calibration which is enough to project a point from image plane to camera frame or inverse. The Equations to project a point Q to image plane is given in Equations 3.31 and 3.32.

$$x = f_x\left(\frac{X}{Z}\right) + C_x \tag{3.31}$$

$$y = f_y\left(\frac{Y}{Z}\right) + C_y \tag{3.32}$$

In Equations 3.31 and 3.32, X, Y and Z denotes the coordinates of the point Q on camera frame. Terms x and y denotes the coordinates of the point q, which is the projection of the point Q on to the image plane. This projection is also illustrated in Figure 3.4.

Second step of the measurement model is to project the point on camera frame O_c to world plane O_w. In order to achieve this a world plane axis have to be defined. At initialization, the world plane is assigned to where the camera plane is and a constant yaw, pitch and roll between the camera frame O_c and the world plane O_w is assumed. The geometry of the two frames at initializations is illustrated in Figure 3.5.

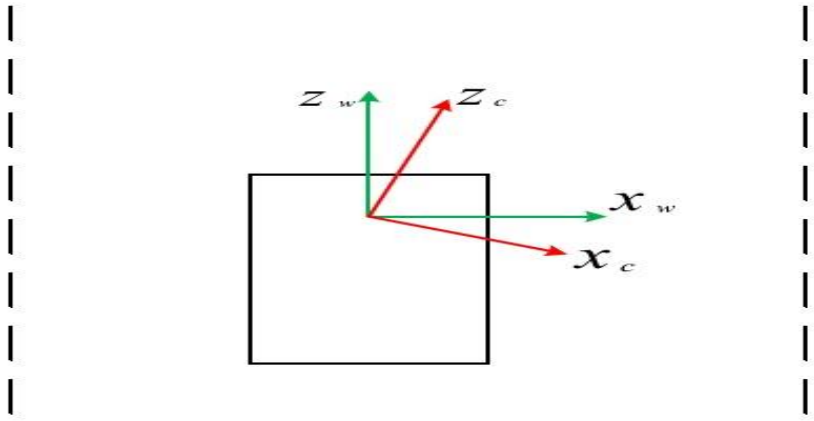

Figure 3.5 : World plane axes and camera frame axes

In Figure 3.5, z_w and x_w denotes the world frame axes and z_c and x_c denotes the camera frame axes. In the world plane, z_w axis is carried with the camera frame z_c axis since we are only interested in lateral localization of the vehicle. This means that world plane (O_w) and camera frame (O_c) origins split from each other only in x-axis

meaning that it only happens when the vehicle moves laterally. In order to project the control points from camera frame O_c to world plane O_w , a translation vector and a rotation matrix have to be defined.

In order to obtain Rotation matrix between the world plane O_w and camera frame O_c , the order of three main rotations (yaw pitch and roll) have to be defined. In this thesis, the order of rotations is defined as pitch, yaw and roll. Because the mounting position of the camera has to be included in the measurement model, this order of rotation is selected. Yaw, pitch and roll values between the world plane O_w and the camera frame O_c is obtained by making a calibration on a known frame. Particular rotation matrices for pitch (β), roll (ϕ) and yaw (θ) are given in Equations 3.33, 3.34 and 3.35 respectively.

$$R_{pitch} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\beta) & -\sin(\beta) \\ 0 & \sin(\beta) & \cos(\beta) \end{bmatrix} \quad \textbf{(3.33)}$$

$$R_{roll} = \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 \\ \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \textbf{(3.34)}$$

$$R_{yaw} = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \quad \textbf{(3.35)}$$

Consistency of order of the rotations is vital for projection, creating a rotation matrix by two different order in multiplications for same yaw, pitch and roll values will not yield to same rotation matrix. The order of rotations are defined while creating the rotation matrix by multiplication of the Equations 3.33, 3.34 and 3.35 as given in Equation 3.36.

$$R_c^w = R_{pitch} R_{yaw} R_{roll} \quad \textbf{(3.36)}$$

In Equation 3.36, the term R_c^w denote that the rotation matrix from world plane to camera frame. The resulting matrix of the Equation 3.36 is given in Equation 3.37, where s stands for $\sin()$ and c stands for $\cos()$. Angles θ , ϕ and β stands for yaw, roll and pitch respectively.

$$R_c^w = \begin{bmatrix} \mathrm{c}\theta\mathrm{c}\phi & \mathrm{c}\beta s\phi + c\phi s\beta s\theta & s\beta s\phi - c\beta c\phi s\theta \\ -c\theta s\phi & c\beta c\phi - s\beta s\theta s\phi & c\phi s\beta + c\beta s\theta s\phi \\ s\theta & -c\theta s\beta & c\beta c\theta \end{bmatrix} \quad \textbf{(3.37)}$$

In order to project the control points from camera frame to world frame , the rotation matrix from camera frame to world plane is required which is defined as the inverse of the rotation matrix from world plane to camera frame. Since rotation matrices are orthogonal matrices, inverse of a rotation matrix is equal to its transpose as given in Equation 3.38.

$$R_w^c = R_c^{w-1} = R_c^{wT} \quad \textbf{(3.39)}$$

A translation vector is required along with the rotation matrix in order to project the control points to world frame correctly. Translation vector is obtained according to the control points extracted from image processing algorithm. The position vectors according to the world plane and camera frame of a control point *P* after a lane change is illustrated in Figure 3.6.

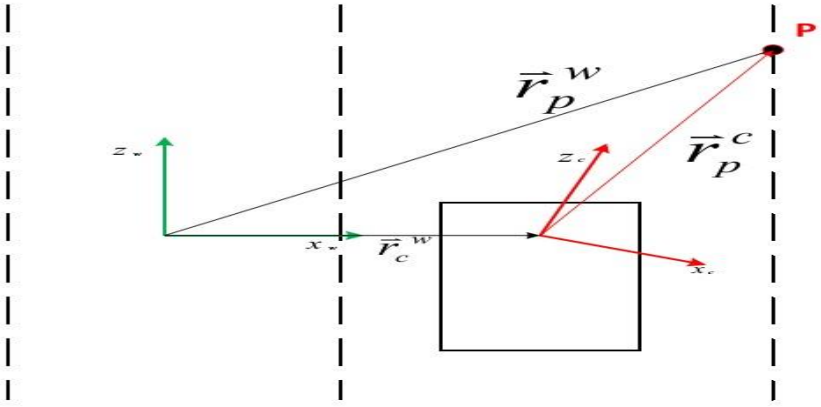

Figure 3.6 : Control point position vectors on camera frame and world plane.

In Figure 3.6, vectors $\vec{r}_p^w$ and $\vec{r}_p^c$ denotes the position vectors according to the world plane and the camera frame respectively. The vector, $\vec{r}_c^w$ denotes the translation vector between the world plane and the camera frame. Position vectors are defined as follows in Equations 3.40, 3.41 and 3.42.

$$\vec{r}_p^c = \begin{bmatrix} P_x' \\ P_y' \\ P_z' \end{bmatrix} \quad \textbf{(3.40)}$$

$$\vec{r}_p^{\,w} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{3.41}$$

$$\vec{r}_c^{\,w} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.42}$$

In the measurement model, projection of the control point's only the u axis on the image plane is considered to simplify the problem. This is applicable since only the lateral position of the vehicle is to be estimated.The measurement model for each control point can be defined as follows:

$$h_i = f_x \left(\frac{P_{i_x}{}'}{P_{i_z}{}'} \right) + C_x \qquad i = 1\,,2\,,3\,\ldots.nx5 \tag{3.43}$$

The terms P'_{i_x} and P'_{i_z} can be obtained with multplication of the rotation matrix R_w^c with the difference of the vectors $\vec{r}_p^{\,w}$ and $\vec{r}_c^{\,w}$ as given in Equation 3.44.

$$\vec{r}_p^{\,c} = R_w^c(\vec{r}_p^{\,w} - \vec{r}_c^{\,w}) \tag{3.44}$$

For the sake of simplicity the rotation matrix R_w^c is denoted as follows:

$$R_w^c = \begin{bmatrix} R_1 & R_2 & R_3 \\ R_4 & R_5 & R_6 \\ R_7 & R_8 & R_9 \end{bmatrix} \tag{3.45}$$

From Equation 3.44 the equations of P'_{i_x} and P'_{i_z} is obtained:

$$P_{i_x}{}' = R_1\left(P_{i_x} - X\right) + R_2\left(P_{i_y} - Y\right) + R_3\left(P_{i_z} - Z\right) \qquad i = 1\,,2\,,3\,\ldots.nx5 \tag{3.46}$$

$$P_{i_z}{}' = R_7\left(P_{i_x} - X\right) + R_8\left(P_{i_y} - Y\right) + R_9\left(P_{i_z} - Z\right) \qquad i = 1\,,2\,,3\,\ldots.nx5 \tag{3.47}$$

The completed measurement model is obtained by inserting Equations 3.46 and 3.47 into Equation 3.43, which is given in Equation 3.48.

$$h_i = f_x \left(\frac{R_1\left(P_{i_x} - X\right) + R_2\left(P_{i_y} - Y\right) + R_3\left(P_{i_z} - Z\right)}{R_7\left(P_{i_x} - X\right) + R_8\left(P_{i_y} - Y\right) + R_9\left(P_{i_z} - Z\right)} \right) + C_x \qquad \textbf{(3.48)}$$

$$i = 1\,,2\,,3\,\ldots.nx5$$

The linearized observation matrix H used in EKF is calculated by the partial derivatives of measurement model with respect to the states. The definition of the observation matrix $H$is given in Equation 3.49.

$$H = \begin{bmatrix} \frac{\partial h_1}{\partial x} & \frac{\partial h_1}{\partial \dot{x}} & \frac{\partial h_1}{\partial P_{1_x}} & \cdots & \frac{\partial h_1}{\partial P_{nx5_x}} \\ \vdots & \vdots & \ddots & & \\ \vdots & \vdots & & \ddots & \\ \frac{\partial h_{nx5}}{\partial x} & \frac{\partial h_{nx5}}{\partial \dot{x}} & & & \frac{\partial h_{nx5}}{\partial P_{nx5_x}} \end{bmatrix}_{(5n)x(5n+2)} \qquad \textbf{(3.50)}$$

The linearized Observation Matrix H is given in 3.50, defining H as the projection allowing us to feed EKF algorithm directly with the extracted control points on the image frame. The structure of EKF calculates the projection and back projection inside its own iteration cycle. The observation covariance matrix R is set to constant value between 50 and 200 after observing the performance of the algorithm. Observation covariance affects the minimum value of the innovation covariance, which is used to create region of interest (ROI) in the input image, extraction part of the system. Thus, setting it low range causes the ROI to get smaller than desired value and setting it high range will result in unwanted features in detection.

3.4 Fusion with IMU

In fusion phase of the system, a sub-module of the IMU and the gyroscope is used to support the computer vision module. IMU that is used in the system has inherent Kalman Filter algorithm embedded in its own DSP, which cannot be reached from outer algorithms however, measurements from sub-modules can be obtained along with the orientation values. The sub-modules are accelerometer, gyroscope and magnetometer. Orientation output of the IMU is not used in the fusion part because of the embedded Kalman Filter, which cannot be modified. Therefore, as a first step of the fusion only gyroscope is fused with the computer vision module.

In order to add gyroscope to the computer vision system explained in sections 3.2 and 3.3, two new states are required because the gyroscope measures angular velocity. The two new states added to the system model are, (i) yaw denoted as θ and (ii) yaw rate denoted as $\dot{\theta}$. Adding these states means that the assumption of the constant yaw between the world plane and the camera frame is no longer valid. The new state vector with the additions is given in Equations 3.51.

$$\underline{x}_{new} = \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \\ P_{1_x} \\ \vdots \\ \vdots \\ P_{5n_x} \end{bmatrix} \tag{3.51}$$

Exactly the same procedure is applied to the system with new states as previously explained in sections 3.2 and 3.3. The definitions of the state update functions of yaw (θ) and yaw rate ($\dot{\theta}$) are given in Equations 3.52 and 3.53 respectively.

$$f_{\theta} = \theta + \left(\dot{\theta} + \gamma\right)\Delta t \tag{3.52}$$

$$f_{\dot{\theta}} = \dot{\theta} + \gamma \tag{3.53}$$

State update functions in Equations 3.52 and 3.53 are derived with regard to constant velocity model, which is used to define the state updates of lateral position x and velocity $\dot{x}$ in Equations 3.21 and 3.22. The term γ refers to the noise in Gaussian profile similar to α inside the lateral position model. The Jacobian of the state update functions according to the new state vector is defined in Equation 3.54.

$$F = \begin{bmatrix} \frac{\delta f_1}{\delta x} & \frac{\delta f_1}{\delta \dot{x}} & \boldsymbol{\frac{\partial f_1}{\partial \theta}} & \boldsymbol{\frac{\partial f_1}{\partial \dot{\theta}}} & \cdots & \frac{\delta f_1}{\delta P_{5n_x}} \\ & \vdots & \vdots & \vdots & \ddots & \vdots \\ & \vdots & \vdots & \vdots & & \vdots \\ \frac{\delta f_{(5n)+4}}{\delta x} & \frac{\delta f_{(5n)+4}}{\delta \dot{x}} & \boldsymbol{\frac{\partial f_{(5n)+4}}{\partial \theta}} & \boldsymbol{\frac{\partial f_{(5n)+4}}{\partial \dot{\theta}}} & \cdots & \frac{\delta f_{(5n)+4}}{\delta P_{5n_x}} \end{bmatrix} \tag{3.54}$$

The bold terms inside the matrix F are the only newly added terms to the state transition matrix defined in Equation 3.24. The noise covariance matrix Q also need

to be updated with regard to the new noise vector. The new noise vector w and the updated G matrix, which is used in Equation 3.26, are given in Equations 3.55 and 3.56 respectively. The calculation in Equation 3.56 yields to the G matrix given in Equation 3.57.

$$w = [\alpha \; \gamma]^T \tag{3.55}$$

$$G = \begin{bmatrix} \frac{\partial f_1}{\partial \alpha} & \frac{\partial f_1}{\partial \gamma} \\ \frac{\partial f_2}{\partial \alpha} & \frac{\partial f_2}{\partial \gamma} \\ \vdots & \vdots \\ \vdots & \vdots \\ \frac{\partial f_{5n+4}}{\partial \alpha} & \frac{\partial f_{5n+4}}{\partial \gamma} \end{bmatrix} \tag{3.56}$$

$$G = \begin{bmatrix} \Delta t & 0 \\ 1 & 0 \\ 0 & \Delta t \\ 0 & 1 \\ 0 & 0 \\ \vdots & \vdots \\ \vdots & \vdots \\ \frac{\partial f_{5n+4}}{\partial \alpha} & \frac{\partial f_{5n+4}}{\partial \gamma} \end{bmatrix} \tag{3.57}$$

Adding a new sensor means that new measurements are introduced to the system, therefore the measurement model also need to be updated. The two new states are defined as yaw θ and yaw rate $\dot{\theta}$. In the measurement model only yaw rate from the two states are included, which is the measurements obtained from the gyroscope. Since the gyroscope directly gives the angular velocity, yaw rate can be added as soon as the velocity measurements are received to the measurement matrix. The new measurement matrix is given in Equation 3.58.

$$H = \begin{bmatrix} 0 & 0 & \boldsymbol{0} & \boldsymbol{1} & 0 & \dots & 0 \\ \frac{\partial h_1}{\partial x} & \frac{\partial h_1}{\partial \dot{x}} & \frac{\partial h_1}{\partial \boldsymbol{\theta}} & \frac{\partial h_1}{\partial \dot{\boldsymbol{\theta}}} & \frac{\partial h_1}{\partial P_{1_x}} & \dots & \frac{\partial h_1}{\partial P_{5n+4}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \\ \frac{\partial h_{5n}}{\partial x} & \frac{\partial h_{5n}}{\partial \dot{x}} & \frac{\partial \boldsymbol{h_{5n}}}{\partial \boldsymbol{\theta}} & \frac{\partial \boldsymbol{h_{5n}}}{\partial \dot{\boldsymbol{\theta}}} & \frac{\partial h_{5n}}{\partial P_{1_x}} & \dots & \frac{\partial h_{5n}}{\partial P_{5n+4}} \end{bmatrix} \tag{3.58}$$

Bold terms inside the measurement matrix H are new terms included because of the new state matrix of the model. The first row is calculated via the partial derivative of the gyroscope measurement model with respect to state vector. First row in the measurement matrix H is the Jacobian of the Equation 3.59 according to the new state vector.

$$h_{yaw\,rate} = \dot{\theta} \tag{3.59}$$

From the previously explained rotation matrix from camera frame to world plane R_w^c definition in Equation 3.36 and the measurement function definition in Equation 3.48, the bold terms inside the measurement matrix H are calculated. The measurement matrix H is linearized in every step of the algorithm around the estimated new states.

EKF algorithm provides some useful information about the states, which can be used to increase the performance of the image processing module. Position estimates of the control points on image plane is used along with the covariance values for each states provided by EKF is used to create an adaptive ROI. Moreover, from the estimated positions of the control points the estimated slope of the lanes on the image can be extracted via simple slope calculation between two points. The orientation of the steerable filters' is updated adaptively by the estimated slope. The estimated slope calculation is given in Equation 3.60 for the two control points $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$.

$$Estimated\ Slope = \frac{y_2 - y_1}{x_2 - x_1} \tag{3.60}$$

The algorithm is initialized with the initial values for the state vector. Initial values are defined by taking, the approximate lane width of the roads where the test data is collected. At initialization, it is assumed that the vehicle is at the center of the lane and each lane width is approximately 4m and the vehicle is at zero position laterally. In addition, initial values for the covariance for control points need to be defined in order to initialize the image processing module. Since the covariance are used to create a ROI in image processing module , at initialization the covariance are set to image width to enable the image processing module to check whole image for lanes. The

diagram relation between the image processing module and the tracking module is illustrated in Figure 3.7.

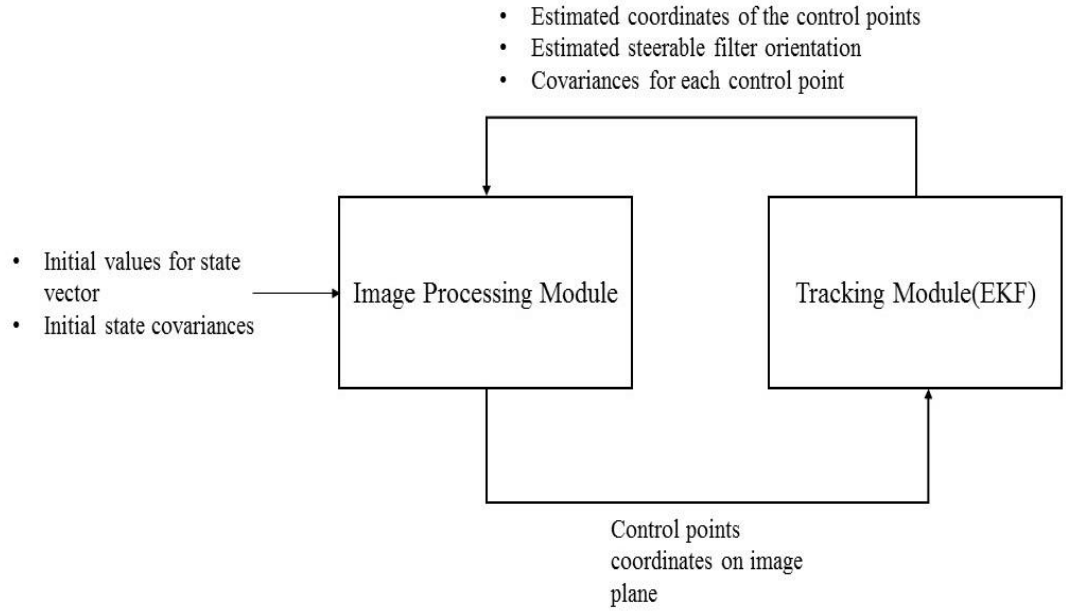

Figure 3.7 : Block diagram of the lane tracking system.

4. EXPERIMENTAL RESULTS

Experiments are made offline on a previously collected data with a test vehicle provided by OTOKAR. Both, the stand-alone computer vision module and computer vision supported with IMU is experimented on same data in order to make a reasonable comparison between the two algorithms. In this section, the collected data and the sensors used are explained in section 4.1, in section 4.2 the experimental results of the proposed algorithm is explained.

4.1 Experiment Data

Approximately 10 hours of synchronous image and IMU data is collected on intercity roads. Data collecting vehicle was OTOKAR M-2010 equipped with a forward-looking Logitech C270 camera attached on the front glass and Xsens Mti-10 series IMU fixed on the center of gravity of the vehicle. Mti-10 is an IMU composed of three sub-modules comprising an accelerometer, a gyroscope and a magnetometer. Along with the IMU output in Euler angles or Quaternions, outputs of sub-modules can be also recorded. Mti-10 can provide raw data and calibrated data from accelerometer, gyroscope and magnetometer based on the application. Additionally, Mti-10 has a built in Kalman Filter fusing the sub-modules to provide a more robust orientation information as sensor output. The collected data consists of gyroscope and accelerometer recordings in 90Hz synchronized with 9Hz image data. Frequencies of both sensors are defined by the maximum reliable frequency of Logitech C270. Even though, C270 is capable of working at 15fps, while recording data through MATLAB, maximum fps with no frame skipping reduces to 9fps. Synchronization is made on software rather than hardware-level because of the webcam limitations. In order to achieve hardware synchronization applying hardware trigger on both sensors is more preferable because it eliminates the additional processing step for synchronization in the software. However, C270 has no input for a hardware trigger; therefore, synchronization is made in MATLAB for data collecting purposes only. This is achieved by the built in buffer of the MTi-10 and MATLAB's frame buffer for

cameras. With initialization, both buffers are triggered to collect 9 samples and 90 samples for the webcam and IMU respectively. Axis of the IMU is illustrated in Figure 4.1.

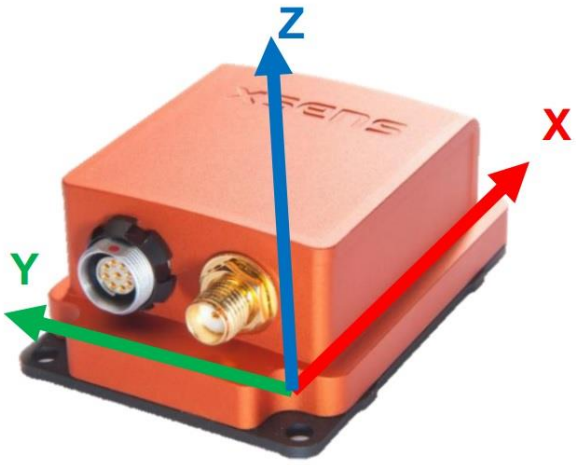

Figure 4.1 : Mti-10 IMU axis [43].

Mti-10 is mounted in such a way so that x axis would be pointing at the direction of the vehicle. The relation between the assigned world plane at the initialization and the Mti-10 axes is demonstrated in Figure 4.2. It can be seen that x_{Mti} axis of the Mti-10 refers to z_w axis of the world plane and y_{Mti} refers to x_w axes of the world plane.

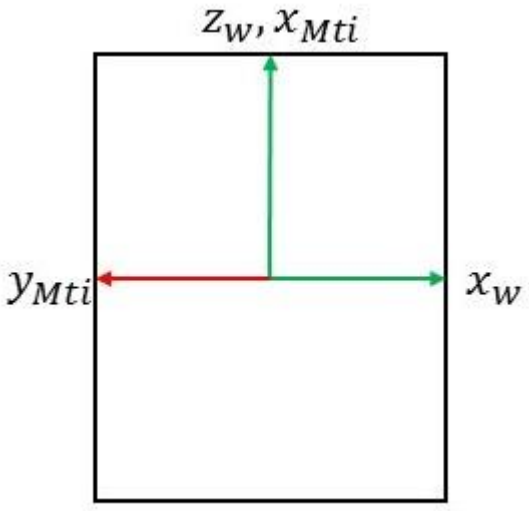

Figure 4.2 : World plane and Mti-10 axes.

The illustration of the synchronized data stream is in Figure 4.3. Every box in Figure 4.3 represents the received data in each time step Δt , it should be noticed that in every 10 received data of IMU, one camera frame is received. Time step of the data stream is defined by Mti-10's measurement period, which is $\frac{1}{90} s$.

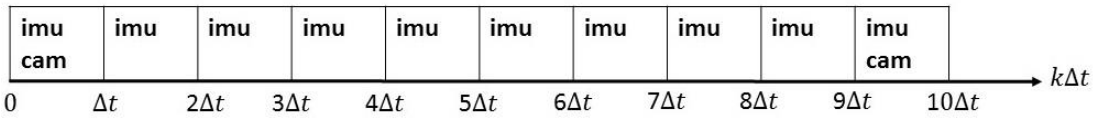

Figure 4.3 : Data stream.

The buffers are triggered every second allowing the buffer of the camera and Mti-10 to hold up 9 frames and 90 measurements respectively. At every second, images are logged into a video object inside MATLAB and the received data from IMU is logged into an array inside MATLAB. Mti-10 provides the data to MATLAB as a vector containing accelerometer and gyroscope measurements, the measurement vector is illustrated in Figure 4.4.

Figure 4.4 : Measurement vector of the MTi-10.

In Figure 4.4, Acc. x, y and z denotes the accelerations in along the x ,y and z axes of the IMU and Gyro x, y and z denotes the angular velocities around x, y and z axes. Accelerations and angular velocities are received are in m/s^2's and rad/s's respectively.

4.2 Experimental Results

The developed algorithm is applied to previously collected data on MATLAB. Some of the built in functions make MATLAB easier to apply the algorithm offline to test main algorithm before it is applied in real-time in C-code or any other lower-level language. While applying the image processing algorithm, 'polyfit' function of the MATLAB which applies Least Squares method to fit a polynomial to data is used. Additionally, image processing toolbox of MATLAB is used to process the input images to be fused with IMU outputs. MATLAB image processing toolbox has built in functions for Canny Edge Detection and Hough Transform which is used in the application. However, all the other parts of the algorithm such as steerable filters, ROI creation and EKF are developed function by function for easy use in the future. Frames on the collected data are marked manually to detect lane positions and it is used for benchmarking different algorithms or measuring the performances of algorithms providing the ground truth for comparison.

In Figure 4.5, successful results of tracking of the control points on image plane are represented. Green points refer to the measurements of the control points obtained from the image processing module and red points refer to the tracking estimations of the control points on the image plane.

Figure 4.5 : Successful results of tracking algorithm.

Once the algorithm solves the problem of tracking the control points referring to the lanes, it is able to track the control points even when no measurements are received from the image processing module of the algorithm as illustrated in Figure 4.6. Therefore, this algorithm is able to work robustly even if the image is not available for a limited period. This situation is quite possible in intercity roads when the image quality is impaired by weather conditions or when the lane marks do not exist

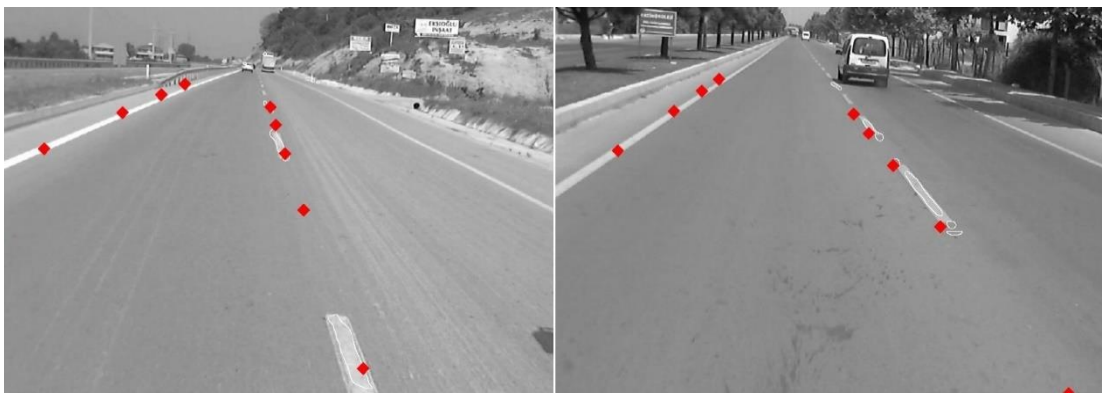

Figure 4.6 : Output of the algorithm when no measurements are received.

In section 2.3 it is mentioned that a first order polynomial is fit the onto feature points corresponding to the lanes. This assumption is made for highway scenarios considering that on highways curvature of the road decreases allowing us to represent the lanes with a first order polynomial up to a distance 20 meters from the vehicle. The results on curved roads are illustrated in Figure 4.7.

Figure 4.7 : Tracking algorithm results for curved roads.

The tracking algorithm fails in various scenarios such as bad initialization or long periods where no measurements are received meaning that there is either no image is received or image processing algorithm is unable to detect any lanes. Initialization is vital for the system performance because, if the tracking algorithm cannot detect any control points regarding to the lanes at initialization it is likely to lose the track of the lanes completely. This happens because of a failure in the estimation of the coordinates, covariance of control points and the steerable filter orientations are used in the image processing module or poor lane markings at initialization. If image processing module is initialized with inaccurate initial values, the image processing module will be unable to detect lane features due to wrong ROI position of the image and wrong steerable filter orientation. In addition to this, when no measurements are received for a long periods of time two cases of failure happens according to the lateral velocity state of the system. First, if the lateral velocity of the system is close to zero before the long period that no measurements are received, the coordinates control points' coordinates stay constant as if vehicle does not move laterally. Second case is when the lateral velocity state is different from zero. In this case, the algorithm estimates as if the vehicle is moving in the direction of the lateral velocity during the period when no measurements are received. Lastly, it may be that the estimates are correct at the initialization; however, image processing module is unable to locate any lane features due to various reasons such as poor lane markings or obstruction by other

road objects or shadows. These cases arise no problems besides; the algorithm can track the lanes when no measurements are received. However, algorithm fails when no measurements are received for consecutive time steps. Failures of the algorithm due to bad initializations are illustrated in Figure 4.8.

Figure 4.8 : Failure cases of the algorithm.

Even though, it is seen that one of the lanes are found at the initialization , the other lane does not exists because the image processing module cannot locate any lane feature.

In Figure 4.9 graph of a test similar to loop closure tests used in SLAM [44] is illustrated. During the 180 frames, the vehicle switches lanes and then switches back to the previous lane. The lane switching process starts at around 45th frame and switching back starts at around 125th frame. Moreover, when the vehicle is switched back to previous lane, the result indicates a value that is very close to zero, which shows that the algorithm localizes the vehicle correctly. Considering this test is made

with a human driver, it is unreasonable to expect the lateral position to be exactly zero at the end of the process. The lateral deviations inside the lane can also be observed.

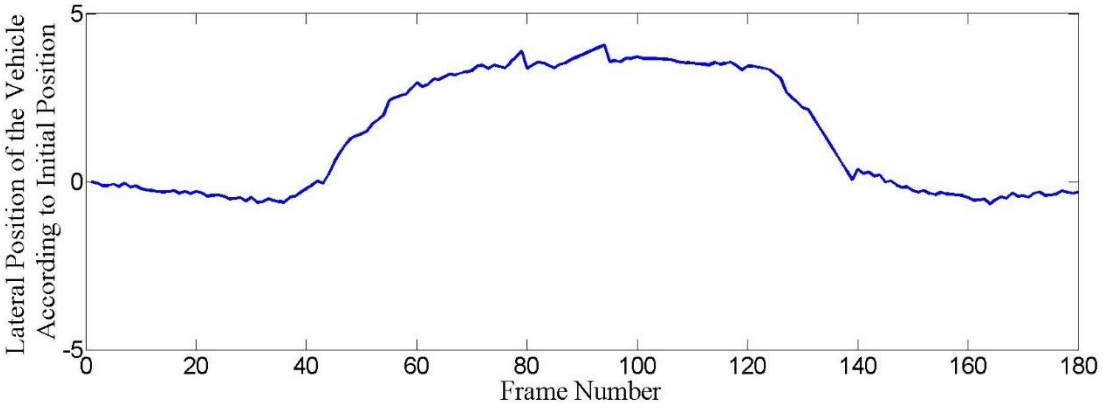

Figure 4.9 : Lateral position estimates of EKF according to the initial position.

Estimated lane slope on image plane is used to create steerable filters in estimated orientations. If the estimated orientations are wrong, steerable filters will be unable to extract lane features for the future steps of the image processing module. Therefore, a comparison of the ground truth, measurements, and algorithm estimates of the lane slope on image plane is represented in figure 4.10.

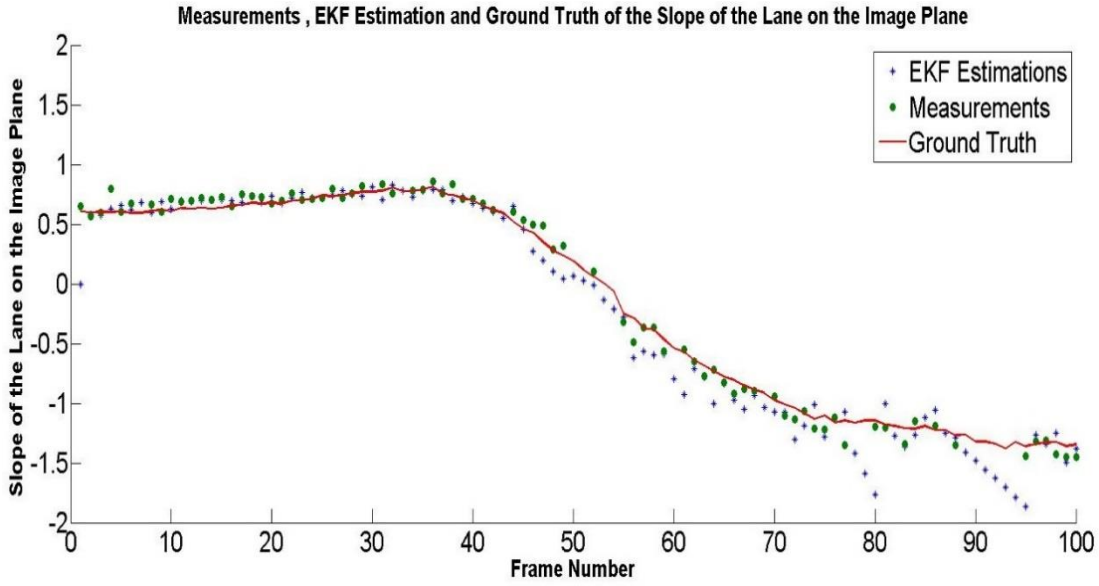

Figure 4.10 : Measurements, EKF estimations and ground truth of lane slope on image plane.

Results indicate that 85% success rate is present considering 0.2 error between the ground truth and estimation as a threshold for success. RMS of the error is calculated as 0.1706. The lane is changing starts around 45th frame. It can be seen that even if there is no measurement exists for consecutive frames and even if the estimation of

the slope deviates from the ground truth, as measurements start to recover the slope is started to be estimated within error tolerance, which is less than 5 frames. This is a tolerable error because, error in the estimated orientations of the steerable filters can be compensated with the tracking algorithm for five 5 frames.

In section 3.2, it is mentioned that even though the constant position model for control points regarding to the lanes is used, tracking algorithm converges control points to their real positions on world plane according to the measurements received. The lateral positions of the control points referring to left and right lanes over 100 Frames are illustrated in Figure 4.11 4.12 respectively.

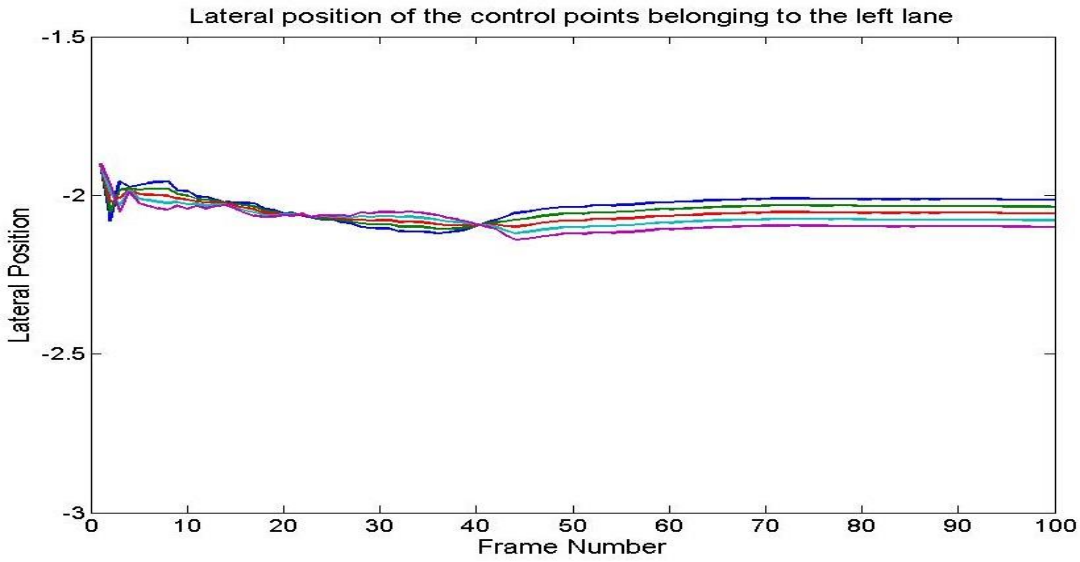

Figure 4.11 : Lateral position of the control points belonging to the left lane.

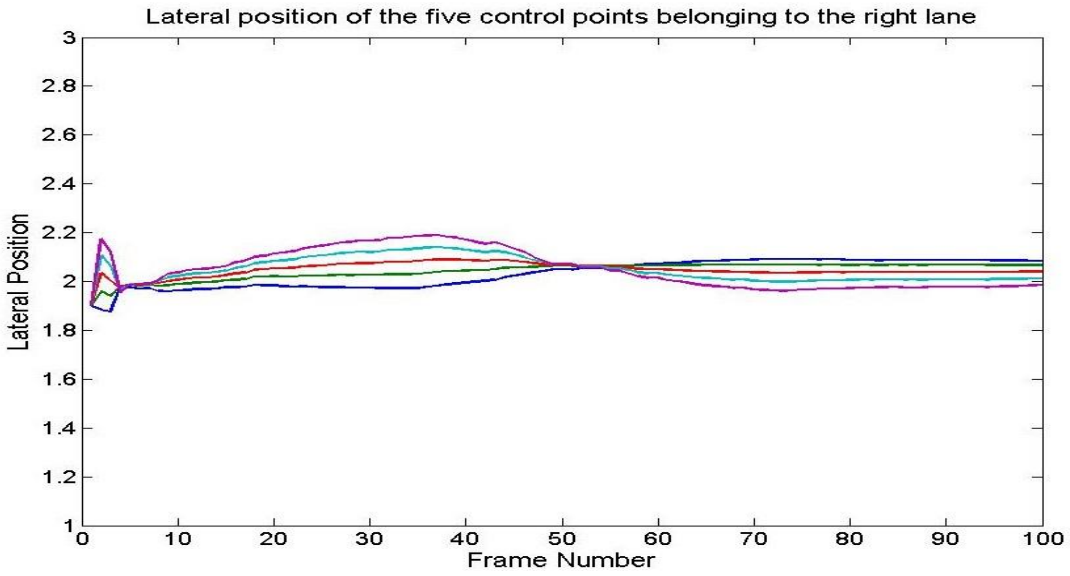

Figure 4.12 : Lateral position of the control points belonging to the right lane.

In Figure 4.11 and 4.12, it can be seen that even though the initialization of the lateral position of the control points are 1.9 meters to left and to right of the initial position of the vehicle , as the measurements received lateral positions of the control points are updated are updated.

5. CONCLUSIONS AND FUTURE WORK

In this thesis work, a multi sensor lane departure warning system is proposed with a mapping and localization approach to the problem. First, existing methods of lane feature extraction, lane modelling, post processing, sensor fusion and tracking used in LDW systems in literature is explained. Then the proposed lane tracking algorithm is explained along with the experimental results.

Proposed lane tracking algorithm is explained in two sections, image processing module and lane tracking. In image, processing the general flow chart of the image processing module is presented. Next, ROI extracting benefiting from the information provided by the proposed tracking algorithm is explained. Presented ROI has an adaptive structure, thus size of the ROI depends on the covariance estimates of the control points. Lane features are extracted by filtering the input image with steerable filters, again the orientation of the steerable filters are adaptively updated by the estimations provided by the tracking algorithm. Steerable filters are applied to the segmented image in order to overcome the perspective problem and Canny edge detection is applied for results that are more robust. Assumptions for the lane structures on the image for highway scenarios are introduced. Then, relying on the assumptions outliers in the extracted lane features are removed by applying Hough Transform and finding peaks in the Hough space for each segment on the image. Least squares method is applied to the found lines via Hough transform in order to fit a first order polynomial representing the lane extracted from the image processing module.

Outputs obtained from the image processing module is fed to the tracking algorithm. In lane tracking section, Kalman filter and its variation for non-linear cases the Extended Kalman Filter is explained briefly. Then the motion model for lane tracking system is introduced along with the state vector. In the motion model, control points, which are used to represent the lanes in pre-determined distances ahead of the vehicle, are explained. Only the lateral position and velocity of the vehicle in addition to lateral coordinates of the control points regarding to the lanes are included in the state vector. Five control points are used to represent each lane for more effective processing. Then the constant lateral velocity model for the vehicle and constant position model for the

control points are presented. The pinhole camera model that is used in measurement model in the tracking algorithm is explained and the assumption on the world plane and the camera frame is introduced. Between the world plane and the camera frame, constant yaw, roll and pitch are assumed in the model. In order to define the measurement model, position vectors for control points are defined for the world plane and the camera frame. Moreover, extracting covariance of lateral position for each control point and the estimated steerable filter orientation is explained which is used to feed the image processing module in the process. An altered version for the proposed measurement and motion model are also explained in order to benefit from the IMU data available. In this altered model two new states, yaw and yaw rate is added to the state vector and constant angular velocity model is applied. By adding yaw and yaw rate to the model, the constant yaw between the world plane and camera frame assumption is discarded. As a first step for the fusion of the IMU, a sub module of the IMU, the gyroscope measurements of the yaw rate is incorporated into the system in the measurement model.

Experiments are made on previously collected data offline. A method for synchronization of the IMU and the camera on MATLAB by exploiting the data buffer provided by the IMU is presented. The structure of the synchronized data stream and the received data vector from the IMU is explained. In addition, the relation between the IMU coordinate system and the world plane assigned at initialization is given. Proposed algorithm is tested on MATLAB, the experiment results contain outputs on the image plane on various scenarios such as curved roads or when no data is received. Experiments on the image processing module with various initial estimates show that poor initial estimates cause the system to fail. In addition, a test similar to loop closure test, which is a commonly used technique in SLAM algorithms, is made. Result of the loop closure test show that the algorithm can successfully estimate the lateral position of the vehicle with an acceptable error when reliable data is received. Another illustrated result is that the estimated slope of the lanes on the image plane, which are used to define steerable filter orientations on the image plane. It is seen that the algorithm can track the slope of the lanes on the image plane even though no measurements are received for consecutive time steps.

Proposed image processing and lane tracking module are very versatile bases for both sensor fusion and testing with other lane feature extraction methods. In addition, it

should be noted that the presented algorithm is more a mapping and localization algorithm rather than lane tracking algorithm in unknown environments. Therefore, the algorithm is a potential sub-module of a future autonomous car.

In future studies, there are several tracks to follow in order to upgrade the proposed system. First track is to upgrade the image processing module of the system in order to obtain more versatile and robust results. This may be achieved by expanding several features of the image processing module. Firstly, introducing a more advanced model for the lanes such as clothoids can be applied. Secondly, a generalized Hough Transform can be derived in order to modify the transform to find parallel curves on the image. Third, a more sophisticated method than the least squares to fit the model to obtained data can be applied. Second track is to focus on the fusion of the IMU's sub-modules to the tracking algorithm. Using, gyroscope, accelerometer and magnetometer together inside the tracking may increase the performance of the system where lanes markings are poor or they may not even exists. Fusing these sensors can be achieved by using a cascaded system of Kalman filters or adding them directly to the proposed tracking algorithm. Third track and last track is to use the same motion and measurement models and apply other tracking algorithms such as Particle Filters instead of EKF and compare the results.

REFERENCES

[1] **World Health Organization**, (2009).World Report on Road Traffic Injury Prevention, Geneva, Switzerland, [Online]. Available: http://www.who.int/ violence_injury

[2] **World Health Organization**, (2013).Global Status Report on Road Safety Available: Available: http://www.who.int

[3] **European Union**, Commission Regulation No 351/2012 of 23 April 2012

[4] **Brugger, F., Burckhardt, M., & Faulhaber,** A. (1989). *U.S. Patent No. 4,861,118.* Washington, DC: U.S. Patent and Trademark Office.

[5] **Lie, A., Tingvall, C., Krafft, M., & Kullgren, A.** (2006). The effectiveness of electronic stability control (ESC) in reducing real life crashes and injuries. *Traffic Injury Prevention*, 7(1), 38-43.

[6] **Abe, T., Hara, M., Kamio, S., Sakita, K., & Takao, M.** (1991). *U.S. Patent No. 5,018,595.* Washington, DC: U.S. Patent and Trademark Office.

[7] **Farr, Glyn PR.** (1990) *U.S. Patent No. 4,966,248*. Washington, DC: U.S. Patent and Trademark Office.

[8] **Vahidi, A., & Eskandarian, A.** (2003). Research advances in intelligent collision avoidance and adaptive cruise control. *Intelligent Transportation Systems, IEEE Transactions on*, *4*(3), 143-153.

[9] **Löwenau, J. P., Bernasch, J. H., Rieker, H. G., Venhovens, P. T., Huber, J. P., Huhn, W., & Reich, F. M.** (1999). Adaptive light control-a new light concept controlled by vehicle dynamics and navigation. *SAE transactions*, 107, 31-36.

[10] **Sotelo, M. Á., & Barriga, J.** (2008). Blind spot detection using vision for automotive applications. *Journal of Zhejiang University SCIENCE A*, *9*(10), 1369-1372.

[11] **Driver Drowsiness Detection** in Wikipedia. Date Retrieved 04.04.2014 http://en.wikipedia.org/wiki/Driver_drowsiness_detection

[12] **Ieng, S., Tarel, J. P., & Labayrade, R**. (2003, June). On the design of a single lane-markings detectors regardless the on-board camera's position. In *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE* (pp. 564-569). IEEE.

[13] **Veit, T., Tarel, J. P., Nicolle, P., & Charbonnier, P.** (2008, October). Evaluation of road marking feature extraction. In *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on* (pp. 174-181). IEEE.

[14] **Broggi, A., & Berte, S.** (1995). Vision-based road detection in automotive systems: A real-time expectation-driven approach. *arXiv preprint cs/9512102*.

[15] **Kluge, K., & Thorpe, C.** (1995). The YARF system for vision-based road following. *Mathematical and Computer Modelling, 22*(4), 213-233.

[16] **McCall, J. C., & Trivedi, M. M.** (2006). Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation. *Intelligent Transportation Systems, IEEE Transactions on, 7*(1), 20-37.

[17] **Muad, A. M., Hussain, A., Samad, S. A., Mustaffa, M. M., & Majlis, B. Y.** (2004, November). Implementation of inverse perspective mapping algorithm for the development of an automatic lane tracking system. In *TENCON 2004. 2004 IEEE Region 10 Conference* (pp. 207-210). IEEE.

[18] **Dong-Joong Kang, Mun-Ho Jung**, (2003). Road lane segmentation using dynamic programming for active safety vehicles, *Pattern Recognition Letters, Volume 24*(16). 3177-3185

[19] **Wang, Y., Dahnoun, N., & Achim, A. (2012).** A novel system for robust lane detection and tracking. *Signal Processing, 92*(2), 319-334.

[20] **Dickmanns, E. D., & Mysliwetz, B. D.** (1992). Recursive 3-d road and relative ego-state recognition. *IEEE Transactions on pattern analysis and machine intelligence, 14*(2), 199-213.

[21] **Gackstatter, C., Heinemann, P., Thomas, S., & Klinker, G.** (2010). Stable road lane model based on clothoids. In *Advanced Microsystems for Automotive Applications 2010* (pp. 133-143). Springer Berlin Heidelberg.

[22] **Jung, C. R., & Kelber, C. R. (2005).** Lane following and lane departure using a linear-parabolic model. *Image and Vision Computing, 23*(13), 1192-1202.

[23] **Ballard, D. H.** (1981). Generalizing the Hough transform to detect arbitrary shapes. *Pattern recognition, 13*(2), 111-122.

[24] **Fischler, M. A., & Bolles, R. C.** (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM, 24*(6), 381-395.

[25] **López, A., Serrat, J., Canero, C., & Lumbreras, F.** (2007). Robust lane lines detection and quantitative assessment. In *Pattern Recognition and Image Analysis* (pp. 274-281). Springer Berlin Heidelberg.

[26] **Guo, C., & Mita, S.** (2009, February). Drivable road region detection based on homography estimation with road appearance and driving state models. In *Autonomous Robots and Agents, 2009. ICARA 2009. 4th International Conference on* (pp. 204-209). IEEE.

[27] **Tang, C. W., Feng, K. T., Tseng, P. H., Chen, C. H., & Guo, J. W.** (2012, April). A pitch-aided lane tracking algorithm for driver assistance system with insufficient observations. In *Wireless Communications and Networking Conference (WCNC), 2012 IEEE* (pp. 3261-3266). IEEE.

[28] **Lim, K. H., Seng, K. P., Ang, L. M., & Chin, S. W.** (2009, August). Lane detection and Kalman-based linear-parabolic lane tracking. In *Intelligent Human-Machine Systems and Cybernetics, 2009. IHMSC'09. International Conference on* (Vol. 2, pp. 351-354). IEEE.

[29] **Meuter, M., Muller-Schneiders, S., Mika, A., Hold, S., Nunny, C., & Kummert, A.** (2009, October). A novel approach to lane detection and tracking. In *Intelligent Transportation Systems, 2009. ITSC'09. 12th International IEEE Conference on* (pp. 1-6). IEEE.

[30] **Apostoloff, N., & Zelinsky, A. (2003, June).** Robust vision based lane tracking using multiple cues and particle filtering. In *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE* (pp. 558-563). IEEE.

[31] **Kim, Z.** (2008). Robust lane detection and tracking in challenging scenarios. *Intelligent Transportation Systems, IEEE Transactions on, 9*(1), 16-26.

[32] **Danescu, R., & Nedevschi, S.** (2009). Probabilistic lane tracking in difficult road scenarios using stereovision. *Intelligent Transportation Systems, IEEE Transactions on, 10*(2), 272-282.

[33] **Freeman, W. T., & Adelson, E. H.** (1991). The design and use of steerable filters. *IEEE Transactions on Pattern analysis and machine intelligence, 13*(9), 891-906.

[34] **Canny, J.** (1986). A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6), 679-698.

[35] **Sobel, I., & Feldman, G.** (1968). A 3x3 isotropic gradient operator for image processing. *a talk at the Stanford Artificial Project in*, 271-272.

[36] **Hough, P. V.** (1962*). U.S. Patent No. 3,069,654*. Washington, DC: U.S. Patent and Trademark Office.

[37] **Duda, R. O., & Hart, P. E.** (1972). Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM, 15*(1), 11-15.

[38] **Haykin, S.** (2002). Adaptive Filter Theory. Assessment.

[39] **Davison, A. J., Reid, I. D., Molton, N. D., & Stasse, O. (2007).** MonoSLAM: Real-time single camera SLAM. *Pattern Analysis and Machine Intelligence, IEEE Transactions on, 29*(6), 1052-1067.

[40] **Welch, G., & Bishop, G.** (1995). An introduction to the Kalman filter.

[41] **Katsuhiko, O.** (2010). Modern control engineering.

[42] **Bradski, G., & Kaehler, A.** (2008). Learning OpenCV: Computer vision with the OpenCV library. O'Reilly Media, Inc.

[43] **Xsens,** Mti-10-series and Mti 100- series user manual.

[44] **Ila, V., Andrade-Cetto, J., Valencia, R., & Sanfeliu, A.** (2007, October). Vision-based loop closing for delayed state robot mapping. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on* (pp. 3892-3897). IEEE.

CURRICULUM VITAE

Name Surname: Barış Özcan

Place and Date of Birth: 18 September 1989 İstanbul

Address: No: 7 Manolya 2/9 56 Ada Ataşehir İstanbul

E-Mail: barisozcan89@gmail

B.Sc.: Bahçeşehir Üniversitesi 2007-2011

PUBLICATIONS/PRESENTATIONS ON THE THESIS

Özcan, B., Boyraz, P., Yiğit, C.B. (2014, July)A MonoSLAM Approach to Lane Departure Warning System. In *Advanced Intelligent Mechatronics (AIM), 2014 IEEE/ASME International Conference.* IEEE. (Accepted)