# Adaptive Observer for Automated Emergency Maneuvers

Fusing cost-efficient onboard sensors with computer vision into a robust estimate of sideslip angle using online covariance calculation

Ruben van Beelen

Master of Science Thesis

Delft University of Technology

Delft Center for Systems and Control

# Adaptive Observer for Automated Emergency Maneuvers

**Fusing cost-efficient onboard sensors with computer vision into a robust estimate of sideslip angle using online covariance calculation**

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Ruben van Beelen

January 2, 2019

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

POLITECNICO
MILANO 1863

DIPARTIMENTO DI ELETTRONICA
INFORMAZIONE E BIOINGEGNERIA

The work in this thesis was supported by Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria.

TU Delft Delft University of Technology

DCSC

# Abstract

One of the most promising ideas in autonomous vehicle control systems is letting the vehicle drive autonomously outside the normal, linear, operating region and letting it "drift". By doing so, the maneuverability of the vehicle could be enhanced. To enable systems that can control this behaviour, estimation of certain vehicle states is needed with high accuracy and high frequency.

In this project, a new solution to this problem is proposed by combining a mixed dynamic-kinematic observer with a single camera that produces velocity measurements based on tracking the ground plane. To improve filtering of the camera velocity measurements, the measurement error covariance matrix is updated online based on a model of the camera measurement error. Evaluation of the new methodology was done on data recorded from a 1:10 scale test vehicle and performance was assessed based on ground truth data obtained using a Motion Capture System.

In normal driving conditions with correctly identified vehicle parameters, an observer without camera still performs better by 25% in terms of RMSE on lateral velocity and sideslip angle estimation. However, the online adaptation of the covariance matrix results in an estimate that is at least 45% more accurate in terms of RMSE than the same observer without online covariance adaptation. Next to that, experiments show that the proposed observer with camera has better robustness to uncertainty in model parameters by almost a factor five in terms of RMSE than the observer without camera.

When the grip of the tires is physically lowered and the vehicle is drifting, the proposed observer can track large sideslip angles ($>30°$), where the state-of-the-art observer without camera is not able. The state-of-the-art observer has an increase in RMSE of 75% on all estimated quantities in comparison to the proposed methodology. These results show that adding a camera to an existing sideslip angle observer greatly enhances robustness of the observer to uncertainty in model parameters and violation of model assumptions. This comes at the cost of losing some accuracy in normal driving conditions.

# Table of Contents

# Preface

This report is the product of twelve months of work on the Thesis Project of the Master of Science program Systems and Control at Delft University of Technology. After initially starting out in the field of Electrical Engineering, the more broad and mathematical program of Systems and Control drew my attention. There I discovered my passion for control systems and vehicle dynamics. In this project I was able to fully dive into these fields and show what I got.

The work done on this thesis has been carried out at Politecnico di Milano and Delft University of Technology. The first six months were spent in Milan, developing the methodology, programming the test vehicle, designing and performing experiments and iterating on the approach. The next six months were spent in Delft processing and analyzing the data, fine-tuning the methodology and creating this report. To anyone considering working on his or her Thesis Project abroad I would fully recommend this.

I would like to thank Matteo Corno for allowing me to come over to Milan and allocating resources to enable this project. Not only was he always ready to give advice and introduce new ideas or concepts to the project when needed, but he also gave enough freedom to design my own experiments.

Next to that, I would like to thank Giulio Panzani for allowing me to come along with him to Università di Trento to perform the experiments and obtain valuable data for the project. He was ready to listen to the daily woes of testing and also provide good company during the trips.

Finally, I would like to thank Hans Hellendoorn for the freedom to carry out the project in Milan and trusting the product I would return with.

# Chapter 1

# Introduction

In the last ten years, driving has become less of an analog, manual task and more of a hybrid and sometimes fully autonomous digital task. Whatever degree of autonomy may be reached in the future, improving the safety and handling characteristics of the vehicle is a task of the past, present and future. The World Health Organisation estimated in 2015 that across the world, 1.2 million people die each year because of road accidents [1]. However, there are strong signs that this number is decreasing. When regarding the European Union only, where there is an above average share of modern vehicles, the number of people killed in road accidents per 1000 inhabitants decreased from 88 to 55 in the last 10 years [2]. In another study conducted within the EU, authors reported that across approximately 10.000 accidents where one or more occupants of the car were severely injured (i.e. no fatalities or light injuries), 40-58 % of the cases reported a loss of control of the vehicle for the driver [3].

The decrease in road accidents leading to fatalities and injuries can partly be attributed to the rise of Advanced Driver Assistance Systems (ADAS) [4]. The first two widely installed and proven systems, Anti-Lock Braking System (ABS) and the Electronic Stability Control (ESC), take sensor data from the car and use that to enhance respectively braking performance and vehicle stability. By taking those measurements, applying mathematics to it and performing a control action, the safety of the driver is enhanced. For example, the overall reduction in accidents due to ESC was estimated to be 14% [5].

The system for stabilizing a vehicle in challenging conditions is the aforementioned ESC. Its main task is aligning the actual yaw moment of the vehicle with the one that is to be expected from the steering angle the driver inputs via the steering wheel. The ESC attempts to stabilize the vehicle by braking one of the four wheels of the vehicle and so avoiding a possible spin. However, there are sources hypothesizing that deliberately entering a spin by an autonomous driving algorithm can enlarge the range of possible paths of the vehicle and thus increase safety in certain scenarios [6].

Unfortunately advanced sensor systems are needed to perform these kind of maneuvers autonomously. The longitudinal velocity $V_x$, as well as the lateral velocity $V_y$ and the angle between those two vectors, the body sideslip angle $\beta$, need to be known at a high frequency with decent accuracy. Current solutions either combine measurements from cheaper sensors, like an Inertial Measurement Unit (IMU) and wheel speed encoders for estimation or they rely on more expensive solutions like a light based radar (LIDAR) or differential GPS (D-GPS) setup [7].

In this thesis a new method for estimating the longitudinal and lateral velocity of a four wheel road vehicle is proposed. The novelty lies in the fact of adding a camera to the existing combination of IMU and wheel speed encoders and filtering the camera measurements in a novel way. This chapter will form an introduction to the later proposed method by first providing the necessary vehicle and tire models. It will then further investigate the subject of driving outside the limits of normal handling and finally discuss the current state of the art in estimation methods. This is all done based on a literature review.

## 1-1 Modelling vehicle behaviour

Before several techniques for estimating the body sideslip angle can be discussed, a general understanding of body and tire models needs to be established. A good starting point for understanding vehicle dynamics is looking at the tires, since these are the only interface between the road and the vehicle. Once this is discussed, a single track model is presented together with some factors influencing vehicle cornering behaviour based on the single track model.

### 1-1-1 Tire models

As said before, the tires are the only interface between the vehicle and the ground plane. The behaviour of a vehicle during cornering depends on the forces the tires can deliver at that moment, which depend on the characteristics of the tire and current state of the vehicle. In this section the relationship between these characteristics and the tire forces is discussed. It is not a full and complete overview of every aspect of a tire, but more a qualitative introduction to vehicle dynamics that will serve as a basis for the next chapters.

#### Generating tire force

The tire is connected to the vehicle via the wheel rim, which in itself is mounted on the front or rear axle. To obtain a longitudinal acceleration, the driver applies torque to the wheels via either braking or accelerating. A speed difference between the tire and body of the vehicle will arise. The common quantity to express this difference, the longitudinal wheel slip ratio $\lambda$ as defined in (1-1) , can be visualized as the ratio of speed difference and the longitudinal vehicle speed with respect to the ground plane:

$$\lambda = \frac{\omega_w R_{\text{eff}} - V_x}{\max(V_x, \omega_w R_{\text{eff}})} \tag{1-1}$$

Where $R_{\text{eff}}$ is the effective wheel radius, $\omega_w$ the rotational speed of the wheel and $V_x$ the longitudinal velocity of the vehicle with respect to the ground plane. The amount of longitudinal

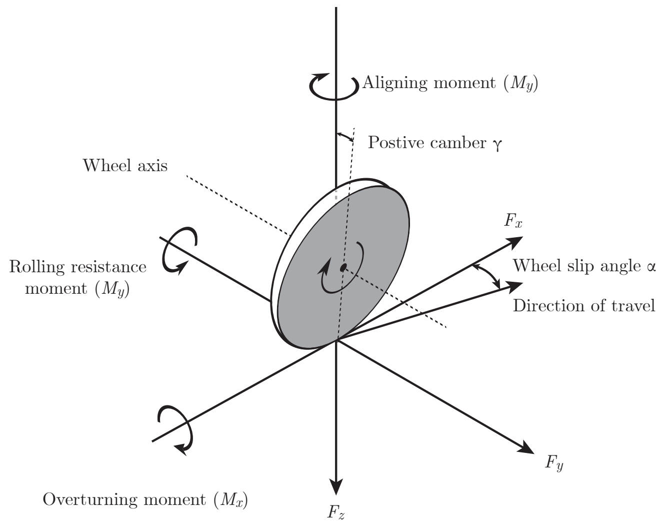


**Figure 1-1:** SAE tire axis system and conventions (simplified), as defined in [8]

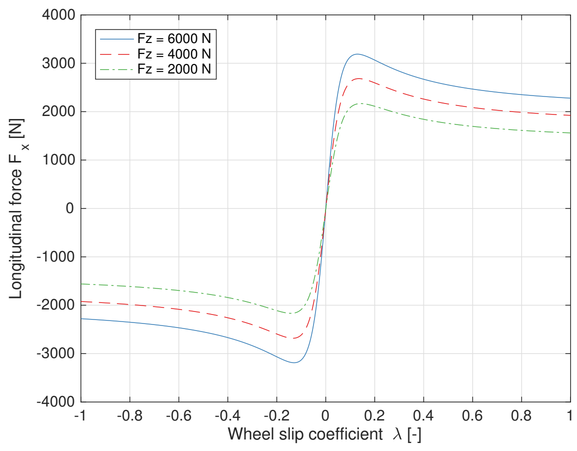
force $F_x$ generated depends on the wheel slip ratio $\lambda$. The relationship between longitudinal tire force and wheel slip ratio for a standard 15 inch tire can be seen in Figure 1-2a. It can be seen that the tire force starts of as an almost linear relationship, then flattens out, reaches its peak value around $\lambda = 0.1$ to then drop off to its final value.

For generating lateral force, the driver of the vehicle can turn the steering wheel, generating a steering angle $\delta$ at all steered axles. When the steered wheels and their tires are turned, the longitudinal axis of the wheel is not aligned with the direction the tire is moving towards, i.e. the direction of the absolute wheel speed. This misalignment is the physical cause of the lateral force a tire generates. The angle between these two quantities (direction of velocity and direction of longitudinal wheel axis) is called the slip angle $\alpha$ and the generated lateral force is dependent on these slip angles. A basic profile for the relationship between $F_x$ and $\lambda$ and $F_y$ and $\alpha$ can be seen in Figure 1-2b. The tire force here also starts off as a linear relationship, to then flatten out to its peak value and slightly drop off after the peak value.

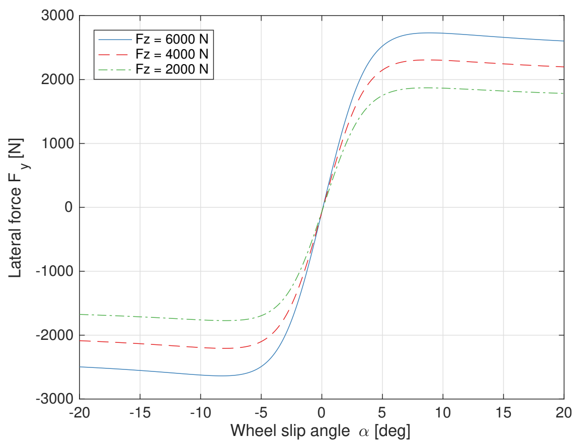
### Parameters influencing tire force

Many different parameters can influence the tire forces that are generated by the tires. Some are linked to vehicle states, for example the force with which the vehicle is pressed down on the road, some are connected to the road the vehicle is driving on and some to the geometry and physical properties of the vehicle. To later be able to discuss drifting and its working principles, only several concepts are needed. The road friction influence, normal force effect and combined slip property will be discussed. Other parameters that have a big influence but are not relevant for this project are the wheel camber angle, tire pressure, toe angle and tire relaxation [9].

**Road friction** The surface the car is driving on has a great influence on the tire forces that can be delivered. The road friction parameter, often called $\mu$, represents the amount of friction the contact surface can deliver to the tire. A rough, dry, asphalt surface can deliver more



**(a)** Relationship between longitudinal tire force and wheel slip coefficient $\lambda$.



**(b)** Relationship between lateral tire force and wheel slip angle $\alpha$.

**Figure 1-2:** Longitudinal and lateral tire force for a 205/60R15 tire at $10\,\mathrm{m\,s^{-1}}$ for three different normal loads using MF-Swift 6.2.0.3

friction than driving on ice. Later in this thesis, the tire friction of the vehicle is reduced by applying plastic tape to the tires.

**Normal force** The second relevant effect influencing tire force is the normal force between the tire and the ground. A tire that is pressed down harder towards the ground plane will be able to deliver more tire force. A practical example of this is Formula 1 cars having massive aerodynamic surfaces. These surfaces enlarge the normal force on the tire by generating negative lift, also known as downforce. The increased normal force on the tire results in improved braking and cornering performance. As can be seen in Figure 1-2, increasing the normal force will lead to greater tire force. However, the increase in force stops at some point and the benefit of having extra normal force will drop off. [10]

**Combined slip** The final factor influencing tire force discussed here is the combined slip property. So far longitudinal and lateral force have been considered separately, but in reality the total force a tire can deliver in one direction is limited by the amount of force it is delivering in the other direction. This can be captured in the following relationship:

$$1 = \sqrt{\left(\frac{F_y}{F_{x,max}}\right)^2 + \left(\frac{F_x}{F_{y,max}}\right)^2} \tag{1-2}$$

The relationship is also referenced as the friction ellipse. An example of this is shown in Figure 1-3. For different combinations of longitudinal slip ratios and slip angles the tire force is calculated. If enough combinations are calculated, the borders of an ellipse start to form. The fact that both forces are coupled places a limit on the possible trajectories for a vehicle. Heavy braking during cornering will lead to a loss of lateral grip and possibly instability, because the tire cannot deliver maximal lateral and longitudinal force at the same moment.

A practical example of this people losing the grip of their (expensive) rear wheel driven car when activating the throttle too early on the exit of a corner and losing grip.

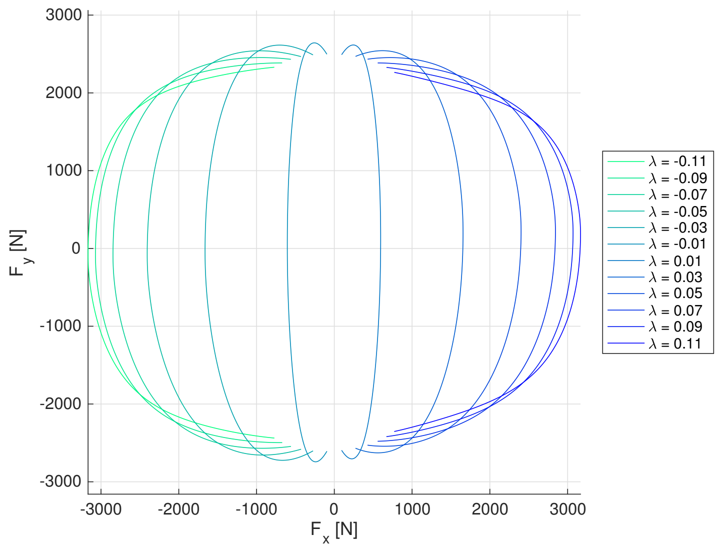


**Figure 1-3:** Friction ellipse for a 205/60R15 tire at $10\,\mathrm{m\,s^{-1}}$, $F_z = 2000N$ using MF-Swift 6.2.0.3. The wheel slip angle $\alpha$ is kept at a constant angle and $\lambda$ is varied at $[-0.3, 0.3]$.

### Practical implementation

Several methods for modelling the tire forces exist, as has been stated earlier. A simple starting point is to express the tire force as a product of the wheel slip angle and a scaling factor:

$$F_{yi} = C_{\alpha_i}\alpha_i \quad i = \{f, r\} \tag{1-3}$$

The scaling factor is often referred to as the cornering stiffness $C_{\alpha,i}$, where i represents the front or rear axle (in case of a single track model) indicated with subscripts $f$ and $r$. The cornering stiffness can either be the linearization of a tire friction curve at the point where $\alpha = 0, \lambda = 0$ or a continuous mapping of the linearization at every point of a tire curve:

$$C_{\alpha_i,0} = \left.\frac{\partial F_{yi}}{\partial \alpha_i}\right|_{\lambda=0,\alpha=0} \qquad C_{\alpha_i} = \left.\frac{\partial F_{yi}}{\partial \alpha_i}\right|_{\lambda,\alpha} \quad i = \{f, r\} \tag{1-4}$$

For more advanced simulation of tire behaviour many different approaches exist. Except for some high fidelity, finite-element like methods, most models are empirical based. Common models are the Dugoff, Fiala and Magic Formula tire models. The Magic Formula itself is derived from test data and fits over 200 parameters. The tire curves presented in this chapter were made using this method, since it's fairly easy implementation and accurate results for this purpose [9]. Later in this project a simpler approach will be adopted to calculate lateral tire force, the hyperbolic tangent model:

$$F_{yi} = \frac{C_i}{k_i}\tanh k_i\alpha_i \quad i = \{f, r\} \tag{1-5}$$

This model calculates the tire force using only one mathematical function. It is a simplified model based on the assumption that there is no combined slip and lateral force doesn't drop off after reaching its maximum value around 5-10° of slip. The main advantage of its easier formulation is that derivatives of the formula can be found in a more convenient way and identification of parameters is easier. Although not documented very often, it is used in some papers [11] and is able to model the tire behaviour in this project sufficiently, as can be seen in Section 5-4.

### 1-1-2 Body models

The next step in modelling the behaviour of a vehicle is introducing a model to study its cornering behaviour. A vehicle consists of four wheels spread over two axles. These four wheels all rotate in one direction and generate tire force, as described in the previous section. For studying the vehicle's stability during a corner, a single track model is often adopted. In a single track model the wheels are lumped together for each axle. Next to that, the model makes a couple other assumptions:

- The road surface is smooth and flat. The vehicle will not experience vertical movement during the experiments.
- Friction is assumed constant over the whole surface.
- Air and rolling resistance is not modelled.
- The steering angle of both front wheels is equal.

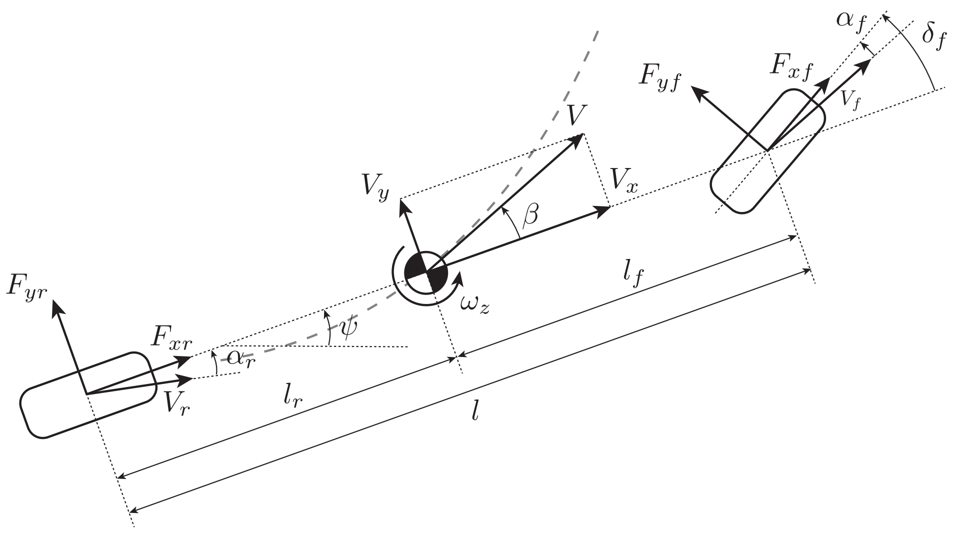


**Figure 1-4:** Overview of the single track model and naming conventions for forces, speeds and angles

The single track model is shown in Figure 1-4 during a left hand corner. When the steering wheel is turned, the front wheels will turn with $\delta_f$. A wheel slip angle $\alpha_f$ will arise at the front axle. The front wheels will generate a lateral force $F_{yf}$ and the vehicle will start to rotate, causing a rotational velocity $\omega_z$ around the $z$ axis and possibly a lateral velocity $V_y$. This angular velocity $\omega_z$ will cause a slip angle $\alpha_r$ at the rear wheel (see (1-9)), which in part will also start to generate lateral force $F_{yr}$. The vehicle will reach a steady state, cornering in the dotted radius with a absolute velocity $V$ tangential to the curve. The absolute velocity $V$ consisting of $V_x$ and $V_y$, with the sideslip angle $\beta$ between $V$ and $V_x$ and the absolute heading of the vehicle is represented with $\psi$. Quantities of interest for the stability of a vehicle during cornering are the lateral velocity $V_y$, the rotational velocity $\omega_z$ and the sideslip angle $\beta$. Assuming a constant longitudinal velocity, they can be modelled as:

$$ma_y = cos(\delta_f)F_{yf} + F_{yr} \tag{1-6}$$

$$I_z\dot{\omega}_z = l_f cos(\delta_f)F_{yf} + l_r F_{yr} \tag{1-7}$$

$$\beta = \arctan\left(\frac{V_y}{V_x}\right) \tag{1-8}$$

The total mass of the vehicle is indicated with $m$, the lateral acceleration with $a_y$, $l_i$ the distance from the centre of gravity to the axle $i$ and $I_z$ the moment of inertia around the $z$ axis. For a single track model with small enough slip angles (maximum of 10°), the slip angles can be approximated with :

$$\alpha_f \approx \delta_f - \left(V_y + \frac{l_f\omega_z}{V_x}\right) \quad \alpha_r \approx -\left(V_y - \frac{l_r\omega_z}{V_x}\right) \tag{1-9}$$

Up until this point, the tire forces can either be a nonlinear model or a linear model, but always depending on $\alpha_f, \alpha_r$ and possibly many more parameters if desired. When replacing the tire force with the product of the cornering stiffness and wheel slip angle, the stability of the vehicle can be investigated further. Combining (1-6) to (1-7) and (1-4) a relationship for the stability of the vehicle during cornering can be derived called the understeer gradient $K_{US}$. The understeer gradient is defined in [8] as:

$$K_{US} = \frac{F_{z,f}}{C_{\alpha,f}} - \frac{F_{z,r}}{C_{\alpha,r}} \tag{1-10}$$

$$\delta_f = \frac{L}{R} + K_{US}\frac{V_x^2}{Rg} \tag{1-11}$$

Where $L$ is the distance between front and rear axle, $R$ the corner radius and $g$ the gravitational acceleration. The understeer gradient can be seen as a balance between the available front and rear lateral grip. Different values for $K_{US}$ result in different cornering behaviour types:

- $K_{US} > 0$: A positive $K_{US}$ will give understeer behaviour, resulting in a vehicle that gives a bigger cornering radius than desired by the driver steering input. The driver will have to increase steering angle to negotiate the turn. Most consumer vehicles are set up this way.
- $K_{US} = 0$: A zero $K_{US}$ will result in neutral cornering behaviour, where the vehicle responds exactly as the driver expects.

- $K_{US} < 0$: A negative $K_{US}$ will result in oversteer behaviour, where the rear of the car "breaks out" due to having poor grip. The vehicle will make a corner with a smaller radius than expected and the vehicle might spin.

The understeer gradient, and especially the oversteer and understeer behaviour give a powerful tool in analyzing vehicle cornering behaviour. Many aspects of vehicle behaviour can be reduced to either affecting the normal load on the vehicle or the cornering stiffness. Two examples will be discussed that are relevant to drifting behaviour.

**Load transfer** The first is load transfer. During longitudinal acceleration, the force distribution between the front and rear axle will depend on the acceleration of the vehicle. A vehicle under braking will have a negative acceleration which will cause the normal load of the front wheels to increase and the load of the rear wheels to decrease. Therefore, the available lateral grip at the front wheels increases and at the rear wheels decreases. Braking in a corner can therefore introduce oversteer in a vehicle [12].

**Combined slip** Another example of effecting the understeer gradient is present in rear wheel drive vehicles. Rear wheel drive vehicles can destabilize the rear of the vehicle, which is in fact shifting to a more oversteer behaviour, by applying rear wheel throttle. The longitudinal force will reduce the total available force for the rear wheels due to the combined slip property. Applying a lot of rear wheel torque during cornering can therefore destabilize the vehicle causing it to oversteer and possibly spin out.

A very basic model of a vehicle is now established. The load transfer, combined slip and many other effects can all be accounted for by extending the model to more degrees of freedom, but right now that is not necessary. A single track model will suffice for the next chapter and implementation in this project.

## 1-2 Cornering at the limits of grip

So far, vehicle behaviour has only been discussed for (quasi-) steady-state cornering behaviour. For a normal consumer vehicle, this will ideally represent 100 % of the driving the car will do. However, there has been a field of research emerging over the last two years into deliberately exciting this type of behaviour and entering the more aggressive maneuvers. A general definition where normal linear driving stops and the nonlinear, unstable drifting starts is when the tire forces saturate and large body sideslip angles come into play. In this section this type of behaviour will be investigated further to know what kind of conditions the desired observer should be able to handle.

For inexperienced drivers driving outside the linear operating region of the tires (see Figure 1-2b) will be an alarming situation and a difficult task. The vehicle is no longer responding in the way the driver expects and controlling the vehicle will become an ambiguous task. Sometimes counter-intuitive driver input is needed to stabilize the vehicle. An infamous example of a situation where this unstable behaviour is actively being used by the driver for better maneuvering capabilities is rally driving.

### 1-2-1 Optimality of drifting

In 2006, [12] was one of the first to combine a behavioral study on rally drivers and a mathematical background to show which principles lie behind the driving style known as drifting. Drifting is an integral part of driving a rally car and is often used to make sharp turns in low-grip conditions, which is exactly the behaviour that seems interesting from a vehicle safety and obstacle avoidance perspective. The key insight into how drivers make drifting maneuvers was that a well-measured destabilization of the rear of the vehicle allows the vehicle to obtain a big rotational velocity. This can be done by either applying rear throttle or front wheel braking, which will both influence lateral grip by respectively the combined slip- and normal load property, as was discussed in the previous section.

After the destabilization and quick rotation, the driver can stabilize the vehicle again by either counter steering, therefore decreasing the slip angle of the front tires and so decreasing front wheel lateral force and balancing the yaw moment, or engaging the throttle to load the rear axle again (in case of a front wheel drive vehicle). Empirical data hints that this kind of maneuver, also called the Trail-Braking maneuver, might be more optimal if the approach speed for the corner is relatively high compared to the corner radius [13]. This might be of great interest for creating emergency maneuver systems that suddenly have to evade obstacles at high speeds in low grip conditions. The Trail-Braking maneuver is also represented in Figure 1-5 .

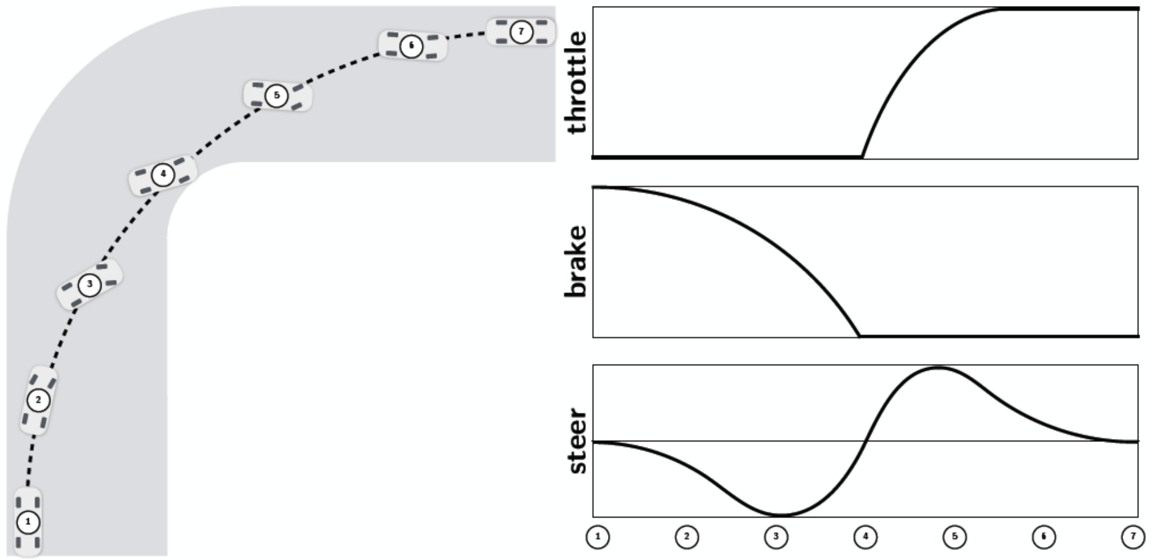


**Figure 1-5:** Overview of the Trail-Braking maneuver, where the vehicle enters a corner (1), the driver unloads the rear axle by braking (2), reaches maximum yaw rate (3), counter-steers (4), applies throttle (5), straightens the vehicle (6) and leaves the corner (7). Image taken from [14]

Next to that, in Velenis et al. [15] it was tested if cornering with a high body side slip angle (i.e. drifting) would result in a lower cornering time when compared to the well-established approach race drivers take for negotiating a turn. Usually race car drivers will try to maximize their cornering speed by braking before the corner and then taking a line through the corner that maximizes their cornering radius but minimizes distance traveled. This allows for larger cornering speeds by reducing the demand for lateral force. [10].

The comparison was done by parameterizing the above described trail-braking behavior and incorporating a single track model with a Magic Formula tire model and combined slip properties. Velenis et al. [15] showed that when driving on low friction surfaces ($\mu = 0.5$) in some cases the drifting maneuver was favorable when optimizing for cornering time. By optimizing not only the cornering time (in fact, minimizing), but also demanding that the vehicle gets its heading changed as quickly as possible the trail-braking maneuver was the quickest. This result was thereafter also verified in CarSim with a high fidelity model.

To top it off, [6] showed that to exploit the full agility of a vehicle, it is necessary to achieve higher sideslip angles, by combining throttle and steering inputs. The physical explanation the authors give for this resides in the tire forces and the delay between them. During a high sideslip maneuver, the front and rear lateral tire force are out of phase and can both contribute to a high yaw moment. This is especially the case when the steering input approaches the minimum acceleration gain region (1-2Hz).

These results together form a strong basis to answer the question *"Why would an automobile want to drift autonomously?"*. Although there is no research that proves that performing a high sideslip evasive maneuver is beneficial to one in the linear operating regime of the tires, for now it is assumed that it is worth investigating this field of research. Now a further look at the mathematical background behind drifting can be taken.

### 1-2-2 Stability of drifting

At almost simultaneous moments, Velenis et al. [13] and Voser et al. [16] shed more light on the dynamics going on behind drifting and managed to sustain a drift for a sustained period around an equilibrium. The approach of [16] will be discussed here. Voser et al. first identified that there exists a set of stable equilibria and a set of unstable equilibria. The equilibria are expressed in body side slip angle, longitudinal velocity, steering angle and yaw rate. Each stable equilibrium represents a steady state cornering motion. By selecting a single track model with a nonlinear tire model, the combinations must satisfy the following equations:

$$\dot{V}_y^* = \frac{F_{y,f}\cos\delta + F_{y,r}}{m} - \omega_z V_x \qquad = f_1(V_y, \omega_z, \delta_f) = 0 \tag{1-12}$$

$$\dot{\omega}_z^* = \frac{l_f F_{y,f}\cos\delta - l_r F_{y,r}}{I_z} \qquad = f_2(V_y, \omega_z, \delta_f) = 0 \tag{1-13}$$

These equations can be numerically solved by selecting a fixed $V_x$ and $\delta_f$ and calculating the tire forces with an identified model. Repeating this process for different combinations of $V_x$ and $\delta_f$ will yield images similar to Figure 1-6. The set of equilibria represented with a solid light gray line corresponds to stable cornering within the limits of handling. The dark grey and dashed light grey lines correspond to equilibria outside the limits of handling, as can be seen by the fact that one of the two tire forces is then saturated. For larger sideslip angles, it can also be seen that countersteer is required to be able to sustain the drift, as was also seen in the empirical data.

In Figure 1-7 the phase portraits for three steering angles (0°,−5°,−15°) can be seen. At low steering angles there exists an unsaturated equilibrium corresponding to normal driving

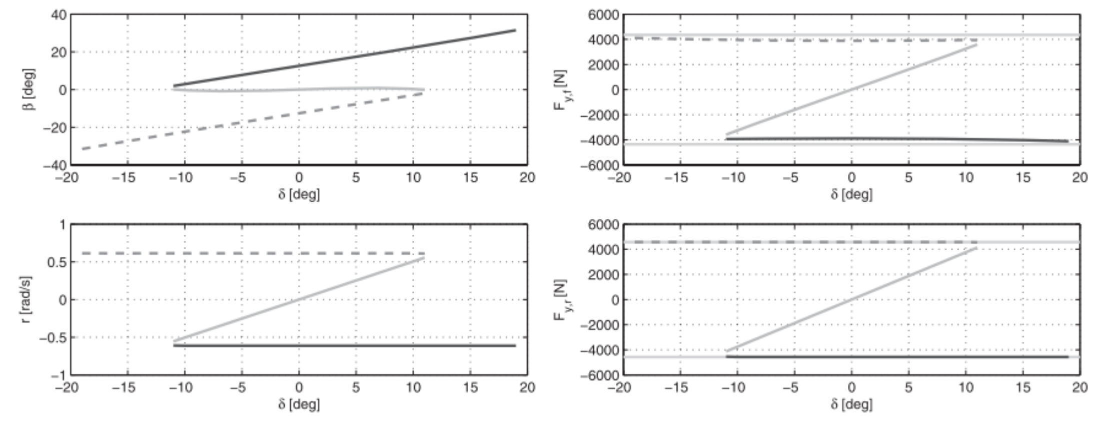
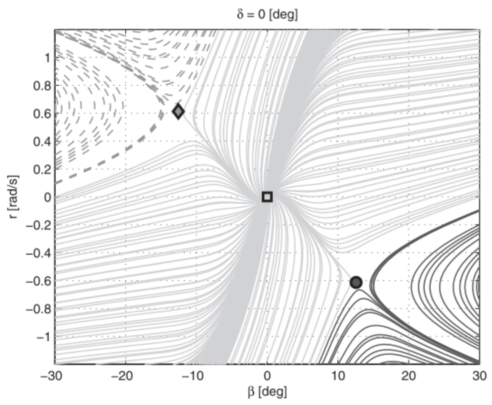


**Figure 1-6:** The equilibria points (left) and the tire forces (right) for the single track model with nonlinear tire forces at $V_x$=8m/s. Images taken from [16]

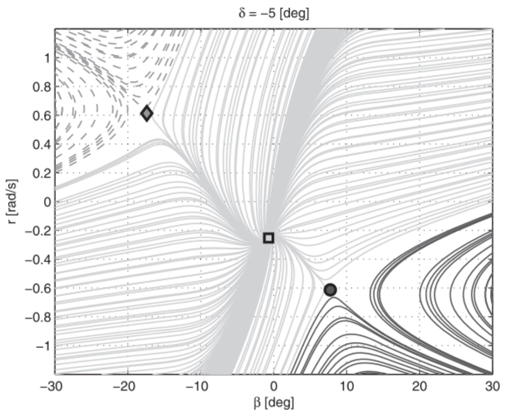
conditions and two drift equilibria. At higher steering angles these equilibria bifurcate and only one unstable, countersteering equilibrium is left.

Finally, some interesting conclusions for the sensitivity of the equilibria are drawn by Voser et al.:

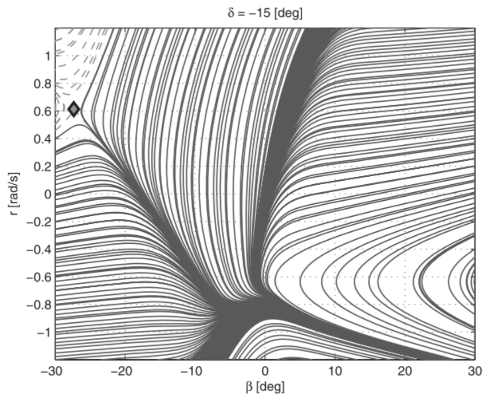
- A $\pm 20\%$ variation in longitudinal velocity will only move the sideslip angle of the equilibrium point by $2.5°$,
- A $\pm 10\%$ variation in friction will move the sideslip angle of the equilibrium by $1.2°$



**(a)** $\delta$=0°



**(b)** $\delta$=-5°



**(c)** $\delta$=-15°

**Figure 1-7:** Phase portraits for different steering angles (0°, -5°and -15°) at $V_x$=8 m/s, light grey lines indicate stable converging trajectories, dashed lines and dark gray lines indicate unstable trajectories. Images taken from [16]

## 1-2-3 Estimation for drifting

Drifting is now described in an empirical and mathematical way, can be modelled and equilibria can be calculated. The next two steps to a complete control system are an observer and a controller for the drift maneuver itself. Controllers for drifting have been discussed and implemented in many different projects and will not be described any further [17, 18, 19, 20, 21]. One important takeaway from these papers is that estimating the sideslip angle with a sufficient frequency and sufficient accuracy is crucial. Since the equilibria are unstable, sampling rates need to be sufficiently high. The biggest remaining problem to be solved for implementing a controller that can perform evasive maneuvers in the nonlinear domain, is an observer that fulfills the needs of the controller.

## 1-3 Estimation of the sideslip angle

Many different ways of estimating the body sideslip angle exist. A vast array of sensors is available for generating measurements that can be put through different types of observers. For a complete overview with more than 100 papers, [22] can be checked. To provide an applicable summary for this project, only methods that employ wheel speed sensors, inertial measurements (angular velocities and planar accelerations) and camera(s) will be considered. These sensors are present on the test platform that will later be used. Next to that, they are also the three types of sensors that are present in consumer vehicles that are hitting the road right now. Other methods, such as using a Differential Global Positioning System (D-GPS) or optical measurement systems are out of question since they are too expensive. This limits the field of options for a possible solution, but enables testing the solution later on the available test platform.

### 1-3-1 Dynamic model based approaches

The biggest part of research on estimating $\beta$ employs an observer that uses a dynamic model of the vehicle, for example the single track model from Subsection 1-1-2. It can predict the behaviour of the vehicle basic on a (simplified) model of it and correct it with measurements. An observer that estimates the sideslip angle based on a dynamic model can be very powerful, as is demonstrated in [23, 24, 25]. It allows for high fidelity models, can deal well with sensor bias and can achieve great accuracy when well calibrated. Effects like roll motion, load transfer, combined slip and other effects can all be modelled and taken into account. Simple dynamic models can already give accurate results, for example in [25]. A three degrees of freedom model is combined with a piece-wise linear fit of the Magic Formula in an Unscented Kalman Filter (UKF) to yield very decent estimates in simulation. However, the authors assumed the road friction coefficient was always known.

Most observers employ either an Extended Kalman Filter (EKF) or an Unscented Kalman Filter (UKF), to be able to capture the nonlinear dynamics of the tire force and sometimes the nonlinear parts vehicle dynamics. In [23] it is shown that the UKF is beneficial to the EKF since it does capture the nonlinear dynamics of the system without linearizing it at every timestep, but choosing the right tuning of the parameters can be tough. Also the difference between the two types of observers was noticable when sampling time became larger (40 ms).

#### Tire force estimation

Both the single and double track vehicle model heavily rely on estimates of the tire forces to produce a good estimate of the lateral velocity and the yaw rate. Two paths of solutions to this problem surfaced and have been improved upon many times.

The first, initiated by Ray et al. [26], proposes to model the whole tire force as a second order random walk process. A three dimensional state space model was used, where the first two states were just integrators and the third state was fed random noise. This was combined with a double track model and put in an EKF. This method can be called Force Estimation. Simulation showed promising results, but no real life experiments were done. Although it is a way where the estimated force can converge to the correct value, estimating the whole tire force as a Gaussian process seems sub-optimal. The only unknown component of the tire force is the amount of available grip from the contact surface. Once that is known, a tire model can be used to calculate the exact amount of tire force according to velocity, steering angle, etc.

This is the starting point for a second method, initiated by Dixon et al. [27]. It tries to estimate the cornering stiffness instead of the whole tire force. This will be called Stiffness Estimation for now. The authors of [27] model the cornering stiffness as a state only influenced by noise and couples it to the tire force in a linear relationship. By assuming that the tire force is a product of the cornering stiffness and the wheel slip angle, the model in (1-14) to (1-18) is obtained by combining a single track model with linear tire force, a kinematic relationship for the longitudinal velocity and extra states for the unknown cornering stiffnesses:

$$\dot{V}_x = \omega_z V_y + a_x \tag{1-14}$$

$$\dot{V}_y = \frac{-2(C_f + C_r)}{mV_x} V_y - \left( \frac{2(C_f l_f - C_r l_r)}{mV_x} \right) \omega_z + \frac{2C_f}{m} \delta_f \tag{1-15}$$

$$\dot{\omega}_z = \frac{-2\left(C_f l_f - C_r l_r\right)}{mV_x} V_y - \frac{2(C_f l_f^2 + C_r l_r^2)}{I_{zz} V_x} \omega_z + \frac{2C_f l_f}{I_z z} \delta_f \tag{1-16}$$

$$\dot{C}_f = 0 \tag{1-17}$$

$$\dot{C}_r = 0 \tag{1-18}$$

The derivative of the cornering stiffness is assumed to only be influenced by noise and not a function. This will make $C_f, C_r$ slowly converge to the real value. In simulation the first results are decent, but the observer converges slowly because the adaptation based on the linear tire model is slow. The authors propose using an extended tire model, although this will increase computational load. Later, several papers were published where the authors have indeed done this. Instead of estimating the linear cornering stiffness, a scaling parameter for a nonlinear tire function is being estimated as a random walk process [28, 29, 30, 31]. In [31] the estimated cornering stiffness is used as a basis of a nonlinear optimization that fits a magic formula to the estimated cornering stiffness.

An important aspect of tire parameter, or tire force, estimating algorithms is that they need to be sufficiently excited to be able to estimate the vehicle parameters. When looking at the observability of the system in (1-14) to (1-18) this is also shown. When the wheel slip angles $\alpha_f, \alpha_r$ are low, the cornering stiffness's $C_{\alpha,f}, C_{\alpha,r}$ become unobservable. When these are not zero, at least the yaw rate and acceleration measurements are required to obtain an observable system [32]. To mitigate this, the authors of [32] use function that adjusts the process noise variance of the states that estimate the cornering stiffness. When driving at the low observability conditions described above, variance is increased of the unobservable states. When performing high transient, high slip maneuvers the variance is decreased. This provides improved tracking of the sideslip angle in low excitation conditions, when compared to the method of Dixon et al. [27].

The dynamic model based observer can provide accurate results when the relevant parameters of the vehicle are well known. This can sometimes flatter the performance of these observers, because of their dependency on good estimates of road friction. When regarding implementation in real consumer vehicles the estimation of these parameters still is a challenging topic.

It requires both sufficiently exciting maneuvers as well as fast convergence of the friction estimation process.

### 1-3-2 Kinematic model based approach

Another approach to estimating sideslip angle is the kinematic model based approach. The kinematic model based observer views the vehicle as a moving and rotating point-mass. The differential equations for the longitudinal and lateral velocity of the body can be seen in (1-19). In this model the longitudinal and lateral acceleration are measured and used as the input signal $\mathbf{u}$. Next to that the yaw rate of the vehicle, i.e. the angular velocity around the z-axis, is also measured and used as a parameter in the $A$ matrix. Therefore, the system is a Linear Parameter Varying (LPV) system of the form $\dot{\mathbf{x}} = A(t)\mathbf{x} + B\mathbf{u}$. This model was first applied for estimating lateral velocity in Farrelly et al. [33].

$$\begin{bmatrix} \dot{V}_x \\ \dot{V}_y \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & \omega(t) \\ -\omega(t) & 0 \end{bmatrix}}_{A(t)} \begin{bmatrix} V_x \\ V_y \end{bmatrix} + \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B} \begin{bmatrix} a_x \\ a_y \end{bmatrix} \tag{1-19}$$

A downside of the system is its observability. The only measurable state is the longitudinal velocity, which can be measured using wheel speed encoders. This assumes there is zero or low longitudinal wheel slip. When assessing the observability of the whole system, the observability matrix $\mathcal{O}$ must be taken into account:

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \omega_z(t) \end{bmatrix} \tag{1-20}$$

The observability matrix $\mathcal{O}$ needs to have rank $n$ for a system of order $n$ and is defined in (1-20). The observability matrix for the kinematic model based observer can aslo be seen in (1-20). The observation can be made that when the vehicle is driving in a straight line, observability will disappear since $\omega_z(t)$ will be approaching zero. Whenever $\omega_z(t) > 0$, the system is observable. In [33] a simple asymptotic observer is proposed of the following form:

$$\begin{bmatrix} \dot{V}_x \\ \dot{V}_y \end{bmatrix} = (A - KC) \begin{bmatrix} V_x \\ V_y \end{bmatrix} + B \begin{bmatrix} a_x \\ a_y \end{bmatrix} + K\omega_z(t) \tag{1-21}$$

$$K = \begin{bmatrix} 2\alpha|\omega_z(t)| & (\alpha^2 - 1)\omega_z(t) \end{bmatrix}^T \tag{1-22}$$

This nonlinear observer does not directly guarantee stability, however it is shown in [33] that the observer is indeed stable. The performance of the observer is validated against two dynamic model based observers in linear and nonlinear handling conditions and holds up well, but is very susceptible to noise and will integrates offsets from the IMU. The sensitivity to noise will mostly be from the fact that the observer has no physical foundation to predict vehicle behaviour based on previous states when measurements contain a large error. It only integrates acceleration and corrects for rotation. When a dynamic model will have noisy measurements for some samples, the algorithm can still do a reasonable guess based on the differential equations it is build on. The second problem for the kinematic model based observer arises when driving on banked roads. Because of the baking the internal y-axis of the vehicle will no longer be aligned with the y-axis of the real world, which is the true direction lateral acceleration will then be pointing towards.

In [34] the kinematic approach from [33] is slightly altered. First, the third term from the original model is changed from $\omega_z(t)$ to $V_x$. Next to that, the observer gain is altered to also feature a term $\alpha_0$, guaranteeing observability of the longitudinal velocity when $\omega_z(t) = 0$. Finally a stabilizing term $F(t) > 0$ is added to the $A(t)$ matrix:

$$\begin{bmatrix} \dot{V}_x \\ \dot{V}_y \end{bmatrix} = \underbrace{\begin{bmatrix} -\alpha_0 - \alpha_1|\omega_z(t)| & \omega_z(t) \\ -(\alpha_2+1)\omega_z(t) & -F(t) \end{bmatrix}}_{A(t)-K_n(\omega_z)C} \begin{bmatrix} V_x \\ V_y \end{bmatrix} + \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B} \begin{bmatrix} a_x \\ a_y \end{bmatrix} + \underbrace{\begin{bmatrix} \alpha_0 + \alpha_1|\omega_z(t)| \\ \alpha_2\omega_z(t) \end{bmatrix}}_{K_n(\omega_z)} V_x \tag{1-23}$$

To compensate for the loss of observability of the lateral velocity, $F(t)$ will be large when the vehicle is driving in a straight line, to force the estimate of $V_y$ to zero. When the vehicle is cornering, $F(t)$ will be close to zero and $V_y$ will be calculated in normal fashion. To detect this straight line behaviour the authors look at several signals. Based on the steering angle, yaw rate and current estimate of side-slip angle $F(t)$ is scaled. Combined with a roll angle estimator to also correct for banking and some logic for the wheel speed encoder readout, a final estimation algorithm is created. The observer is verified on real world experiments and able to track a sideslip angle with a Root Mean Square Error (RMSE) of $e_\beta = 0.78°$ and maximum error of 6.73° in a 4000s test on a low adherence-surface.

A final way to improve the original kinematic model based observer is proposed in [35]. By employing load sensing bearings, the measurement vector $\mathbf{u}$ is no longer the directly measured acceleration, but the acceleration calculated by the tire forces and vehicle mass. Together with heuristics to mitigate the loss of observability during straight driving, similar to [34], good tracking of $\beta$ is achieved on real data up to 15°.

It can be concluded that the kinematic model based observer provides a promising starting point. It uses sensors already present on road vehicles and is able to operate under all driving conditions. However, sensor quality is very influential on the quality of the estimate and a lack of observability when driving in a straight line are still troubling aspect of the approach.

## 1-3-3 Hybrid model based approaches

After the initial proposal of the kinematic model based approach, several hybrid estimation methods surfaced. These methods have in common that they take some part of the dynamic vehicle model, most often a nonlinear tire model and a single track vehicle model, and combine that with the kinematic model based approach. Most of these approaches are backed up with mathematics and a simulation, although some also with real life experiments. In this section, some approaches that are proven with real life experiments are discussed.

Some of the first approaches to combine the method of [33] with some dynamic model are found in [36, 37]. In [37], the authors propose a method where once the yaw rate $\omega_z$ is above a certain threshold, the full kinematic model is applied. If the yaw rate gets below the threshold, the model is reduced to:

$$\hat{V}_x = V_x^m \tag{1-24}$$

$$\hat{V}_y = \omega_z V_x^m + \hat{a}_y \tag{1-25}$$

Where $\hat{a}_y$ is calculated using (1-28) with a linear tire force model and $V_x^m$ is the measured longitudinal speed according to wheel speed encoders. At the time of research, 2004, the $a_x$ measurement was not available in commercial vehicles and therefore the discrete derivative of the wheel speed sensors was used. The authors show the approach can outperform dynamic observer approaches, but is sensitive to noise on the $a_x$ measurement and very sensitive to road banking. In lower grip conditions the hybrid kinematic approach holds up well, even when the tire model is not adapted. This is due to the fact that the calculated lateral acceleration correction only comes in to play when driving straight ahead (when $F_y$ is small) and not during transient maneuvers.

A second example of a hybrid approach is found in [38], where direct integration of the sideslip angle according to kinematic formulas and a small angle approximation is run in parallel with a single track dynamic model observer also estimating the sideslip angle and an online linear cornering stiffness estimation. The authors obtain the kinematic sideslip angle by:

$$\dot{\beta} \cong \frac{a_y}{V} - \frac{a_x}{V}\beta - \dot{\omega}_z \tag{1-26}$$

$$\hat{\beta} = \int \left( \frac{a_y}{V} - \frac{a_x}{V}\beta - \dot{\omega}_z \right) dt \tag{1-27}$$

Due to the direct integration of beta and not applying some (Kalman) observer, the measurements need to be high passed filtered to prevent integrating the signal drift. This makes the kinematic sideslip angle only useful for transient maneuvers. However, the measured quantities $a_x, a_y, \omega_z$ will likely also contain noise, so high-pass filtering seems suboptimal. To also be able to operate during quasi steady-state maneuvers, a fuzzy logic function determines if the kinematic $\beta$ is needed or the one obtained from the single track dynamic model observer. Finally, the estimated linear cornering stiffness is a moving average over the last 10 samples that is updated when the kinematic estimation is active. The authors receive good tracking of $\beta$ on dry asphalt and snow during double lane change maneuvers. The sideslip is ranging from $[-6°, 6°]$. However, when the kinematic part of the observer is active for longer periods of time the signals still seem to suffer from integrating the drift of the IMU.

In [29] the idea for using the kinematic model to estimate sideslip is combined with the Stiffness Estimation seen in some dynamic model based observers. The novelty is that in [29] not a linear cornering stiffness is estimated, but a parameter that scales a complete nonlinear tire friction curve up or down. By applying the assumption that the change in yaw rate is low, the estimated lateral acceleration can be calculated if an updated model of the tire forces is available:

$$\hat{a}_y = \frac{1}{m}\left(F_{yf}(V_x, V_y, \omega_z, \delta_f)\cos\delta_f + F_{yr}(V_x, V_y, \omega_z)\right) \tag{1-28}$$

The authors then continue to define a new observer that is partly based on the kinematic model, but instead of having the measured yaw rate in the observer, it creates a separate state for the yaw rate based on the estimated lateral tire forces and corrects this with the measured yaw rate. Next to that, the tire curve scaling parameter is the fourth state. In the end, an observer is synthesized that is able to follow lateral velocities up to $4\,\mathrm{m\,s^{-1}}$ at a longitudinal velocity of $20\,\mathrm{m\,s^{-1}}$ in changing road conditions. However, the performance on banked roads is not tested. This will introduce extra lateral forces on the vehicle and negatively affect the estimation performance.

Finally [11] seems to be able to take the best of the above described approaches and still maintaining low complexity. The model in (1-19) of [33] is used in combination with the calculated lateral acceleration correction in (1-28) of [29]. This results in a hybrid state observer of the following form:

$$\begin{bmatrix}\dot{\hat{V}}_x\\ \dot{\hat{V}}_y\end{bmatrix} = \begin{bmatrix}0 & \omega(t)\\ -\omega(t) & 0\end{bmatrix}\begin{bmatrix}\hat{V}_x\\ \hat{V}_y\end{bmatrix} + \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}a_x\\ a_y\end{bmatrix} + \begin{bmatrix}k_x & 0\\ 0 & k_y(t)\end{bmatrix}\begin{bmatrix}V_x^m - \hat{V}_x\\ a_y - \hat{a}_y\end{bmatrix} \tag{1-29}$$

Where $V_x^m, a_x, a_y$ are measured quantities, $\hat{V}_x, \hat{V}_y, \hat{a}_y$ are estimated quantities with the latter calculated using (1-28). All velocities and accelerations are time varying, but the $(t)$ is omitted. What is new in this approach, is the simpler $k_x$ and $k_y(t)$. The first is just a constant and no longer dependent on $\omega_z(t)$, the latter is a time varying parameter. This parameter determines the contribution of the calculated lateral acceleration correction. During straight ahead driving, when the observability of the kinematic model would disappear, the contribution of the lateral acceleration correction is set to a high value. When either the sideslip angle or the longitudinal wheel slip becomes large, the parameter is scaled down to a low value using a 2D look-up-table. The calculated lateral acceleration then no longer is representative since the zero yaw rate acceleration is no longer true. For the tire forces in (1-28) a hyperbolic tangent model of the following form is used:

$$F_{yi} = \frac{C_i}{k_i}\tanh k_i\alpha_i \quad i = \{f, r\} \tag{1-30}$$

$C_i, k_i$ are parameters that need to be identified and depend on the road surface. In [11] no online estimation of these parameters is done. However the observer still gives good results in the domain it is designed for, $\beta = [-7°, 7°]$. .

Combining a kinematic model with some parts of a dynamic model based approach can yield good results. The most interesting aspect seems to be taking the calculated lateral acceleration and using that to combat the loss of observability and signal drift problems of the kinematic observer.

## 1-3-4 Camera based approach

The previously described observers all use inertial measurements and wheel speed data to make an estimate of the sideslip angle. However, some examples exist adding computer vision to this mix. Measuring the speed of a vehicle based on the image of a camera mounted in the vehicle is not a new idea [39, 40]. The basic principle is the detection of interesting points in one frame, following those to the next frame by optical flow or feature matching and then calculating the vehicles motion based on the movement of the pixels between two frames. The vehicles motion is often referred to as the *egomotion*. This is a very short description of the algorithm and does not fully do it justice. In Chapter 3 all steps of a Computer Vision Algorithm (CVA) producing vehicle speeds measurements will be discussed.

Nowadays, advanced implementations of the previously mentioned technique exist. The authors of [41] employ extra mathematics that reduce noise and prevent violation of made assumptions to the scene. Usually these algorithms are used for visual odometry, the estimation of the vehicles path based on an image sequence. This is a problem that can be run at a lower frequency ($\sim 5\,\mathrm{Hz}$) without introducing too much error. For sideslip angle estimation and control, higher frequencies are desired. Although these algorithms can calculate the optical flow between two images quite accurately, few examples have been found where a purely computer vision based algorithm is also used for the observation of the sideslip angle of a vehicle.

One of the few examples can be found in [42]. There, an algorithm using images from a single camera mounted near the windscreen of a vehicle tracks objects on the ground plane in front of the vehicle between frames. By assuming all objects are on the ground plane the geometry of the scene can be exploited to estimate the speed of the camera based on the movement of the objects. By knowning the speed of the camera the speed of the vehicle can be calculated too. The algorithm can track the sideslip angle on large sideslip maneuvers up to $40°$ with an average deviation of $5°$ and on small sideslip maneuvers with $\beta \approx 2°$ with an error of $1°$. Well distinguishable (and so well trackable) objects were placed on the road surface. Whenever these objects would disappear, tracking was not possible and the estimation was interrupted. The method looks promising, but there are some big risks to the approach that need to be mitigated. In [43] this approach is repeated on a stereo camera setup and results are better. It is even possible to estimate all three translational and all three rotational velocities. Unfortunately there is no calculation of sideslip angle done in this paper and therefore it is hard to judge the real performance of the method.

There exist some examples in literature where inertial measurements are used to circumvent the interruption of measurements when no objects can be tracked. The research group of Wang et al. [44, 45, 46] have developed an observer that uses a camera as a sensor for the lateral position of the vehicle in comparison to the desired trajectory. The deviation from the desired path, assumed to be parallel to the white lane markings on highway roads, is measured by looking at the deviation angle from the lane markings on the road at a fixed preview distance $l_{pre}$. See also Figure 1-8, taken from [44]. The deviation from the ideal path is called $y_l$. The authors derive a differential equation that couples the vehicle dynamics from a linear single track model to the path deviation $y_l$:

$$y_l = y_{cg} + l_{pre} \sin\psi \approx y_{cg} + l_{pre}\psi \tag{1-31}$$

$$y_{cg} = V_{cg} \sin(\psi + \beta) \tag{1-32}$$

$$= \frac{V_x}{\cos\beta} \sin(\beta + \psi) \approx V_x(\beta + \psi) \tag{1-33}$$

$$\dot{y}_l = V_x\beta + l_{pre}\omega_z + V_x\psi \tag{1-34}$$

(1-34) is then added to a single track model with linear tire force and the measurement $y_l$ coming from the camera is used as a measurement signal to correct the prediction of it in a Kalman filter. The overall observer is accurate and its observations were validated on real life experiments, but only on very small sideslip angles of $\beta = [-1.5°, 1.5°]$. In [46] another aspect of combining image data with inertial measurements is touched upon. Processing image data takes time, between 20-80 ms for their algorithm, and image capturing is often limited to just 30 frames per second in current consumer vehicle cameras. The authors therefore propose a multi-rate Kalman filter with an extra state for the delayed signal from the Computer Vision Algorithm. The multi-rate architecture allows for a smaller observer to run at a higher frequency and once the image data is available, extend to a bigger observer that can deal with image data too.

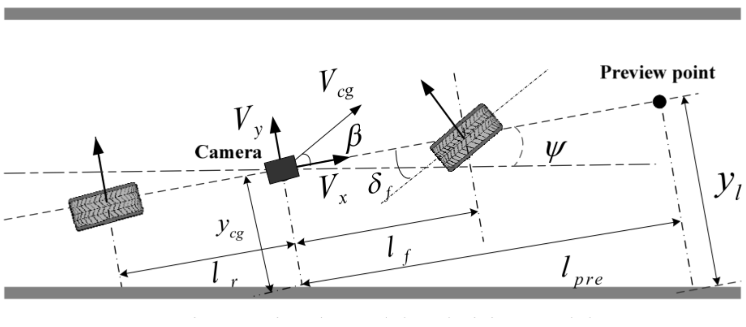


**Figure 1-8:** The look ahead error defined by Wang, 2014. Image courtesy of [46]

The method of Wang et al. is interesting, but the CVA is not based on tracking between frames but on recognizing white lines and measuring the offset between the vehicles heading and the detected line. When driving on a road with a homogeneous color, this cannot be done. This seems to be a problem that is hard to mitigate for the chosen methodology. However, the considerations regarding computation time and multi-rate filters are interesting.

A final piece of work to consider is that of Kuyt [14]. This was made by another TU Delft Master of Science student at Politecnico di Milano. A kinematic model based observer is combined with vehicle speed measurements coming from the camera. The measurements from the camera are based on optical flow, but tracking is aided by placing red markers on the ground. First results from the work are promising, but validation remained to be done on a bigger set of experiments to validate its robustness and effectiveness. This project will continue the work laid out in that project and improve upon the method together with a validation of it.

## 1-4 Conclusion

Automation of the driving task can greatly enhance vehicle safety, as was stated in the previous introduction of this chapter. Letting those autonomous driving algorithms also drive outside the linear operating regime of the tires still remains one of the most promising fields of vehicle dynamics control. Of all parameters needed for control, the lateral velocity and sideslip angle can still not be measured directly in an affordable way. These two parameters are crucial for efficient and accurate control. Dynamic model based methods still rely on estimation of tire friction parameters, which requires sufficiently exciting maneuvers and time to converge. Kinematic model based methods do not suffer from these problem but require clean and drift-free sensor signals, which are seldom present in consumer vehicles. A hybrid approach, combing the best of both solution paths seems the current way to go in the field of sideslip angle estimation. There are signs that adding a camera to this approach can yield favorable results, but validation on real life experiments still needs to be done. Also the camera measurements can provide good estimates, but a lot of factors can instantly render the camera measurements useless. Adequate filtering of these measurements is required.

### 1-4-1 Thesis Objectives

In this thesis the feasibility of adding a camera to a hybrid observer for estimating the sideslip angle is investigated. The final observer should be able to produce more robust estimates of the vehicle's velocities and sideslip angle than a system without camera. The hypothesis before the project is that the camera can aid estimation when vehicle parameters are not identified correctly or the vehicle is undergoing some extreme motion. These are all moments when the model assumptions of the hybrid observer are no longer valid. However it is also expected that adding a camera may reduce accuracy in normal driving conditions, where current algorithms already are quite good. The project is divided in the following steps:

- Design a routine to capture images from a camera and estimate vehicle velocity based on the images.
- Process and filter the camera velocity data so it can be used for estimation.
- Design a methodology to combine the camera velocity data with sensor data and produce an estimate of longitudinal velocity, lateral velocity and sideslip angle.
- Design an experimental setup where the performance of the algorithm can be compared to ground truth data.
- Analyze the performance of the observer with camera data versus a hybrid observer without camera data and analyze the strengths and weaknesses of adding a camera to the system.

### 1-4-2 Thesis Outline

This report will discuss all the steps that were taken to produce the hybrid observer with camera measurements. In Chapter 2 a high-level overview will be given of the hybrid observer that is implemented. It will serve as a basis for the next two chapters and help putting certain parts of the algorithm in perspective. Then, the Computer Vision Algorithm (CVA) will be discussed in Chapter 3. A novel way to filter the CVA estimates, Dynamic Covariance Estimation (DCE), will be discussed in Chapter 4. After that, auxiliary parts of the observer to allow implementation are discussed combined with the system identification in Chapter 5. Finally, experiment setup, filter tuning and results of the experiments are presented in Chapter 6. This thesis is concluded by Chapter 7, in which a summary, discussion and recommendations are presented.

# Chapter 2

# Observer design

In this chapter a high-level overview will be given of the chosen methodology for estimating the body sideslip angle. First an overview of the chosen methodology will be presented in the form of a block-diagram. Then all sub-parts of the methodology will be discussed. This consists of a discussion of the sensors on the available test platform, the estimation algorithm for the vehicle velocities and a separate roll angle observer for camera pose correction.

## 2-1 Observer overview

To give some structure to this chapter and also some support for the reader of this document, a top-down overview of the chosen methodology is presented in this section. In Figure 2-1 an overview of the chosen methodology is given. At the core of the system lies the velocity estimation. This part should take available sensor signals, the camera velocity data and if desired other signals and output a final estimate of longitudinal velocity, lateral velocity and sideslip angle, respectively $V_x, V_y, \beta$.

The three available sensors are a camera, an Inertial Measurement Unit (IMU) and a wheel speed encoder. These are all the sensors that are present on the scaled test vehicle, the Berkeley Autonomous Race Car (BARC). The IMU and encoder signal can be directly used in the velocity estimation. The camera signal, which just consists of images, needs to be processed using a Computer Vision Algorithm (CVA) before it can be fed into the velocity estimation. The CVA relies on an estimate of the current roll angle of the vehicle, produced by the roll estimation block. Finally the Dynamic Covariance Estimation (DCE) calculates weights that the velocity estimation needs to better filter the camera measurements.

The overall goal of this project is to create an observer that can estimate the longitudinal velocity, lateral velocity and sideslip angle. Any methodology should have these quantities as a final product. This can be seen in the block on the right side. This block-diagram forms the backbone of the chosen methodology. In the next section each of the blocks will be explained and in Section 2-6 a complete and more elaborate version of the block-diagram will be presented.

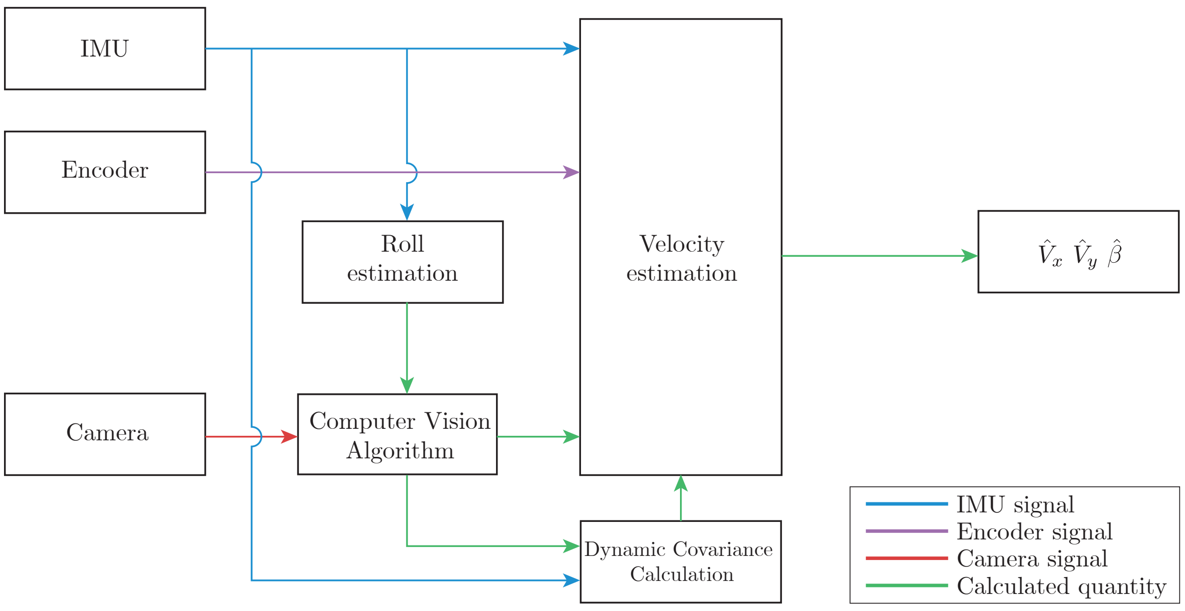


**Figure 2-1:** High-level overview of the chosen methodology for the sideslip angle estimation

## 2-2 BARC platform

The BARC is a modified 1:10 scale radio-controlled (RC) vehicle. It is developed at the University of California, Berkeley, to create a platform for quick, low-cost, repeatable prototyping of autonomous driving systems [47]. The original consumer RC vehicle is fitted with a small form factor computer (Odroid XU4) as main control unit, a microcontroller (Arduino Nano) for analog and serial communication and several sensors. The BARC features a nine degrees of freedom IMU, a 2 megapixel camera and front- and rear-axle incremental encoders for axle-speed. For detailed specifications of the vehicle hardware, see Appendix A-1.

The power of the BARC project lies in the fact that all components can be bought for under $500 at a normal web store. Code can be shared on online platforms such as GitHub and due to the fact that every research group has the same setup, repeatability of experiments is large. To maintain this repeatability, the BARC platform is not modified in terms of hardware for this project. On one hand the final estimate of the sideslip angle might not be optimal since there is no free choice of sensors, but it does serve as a good test for versatility and repeatability of the observer.

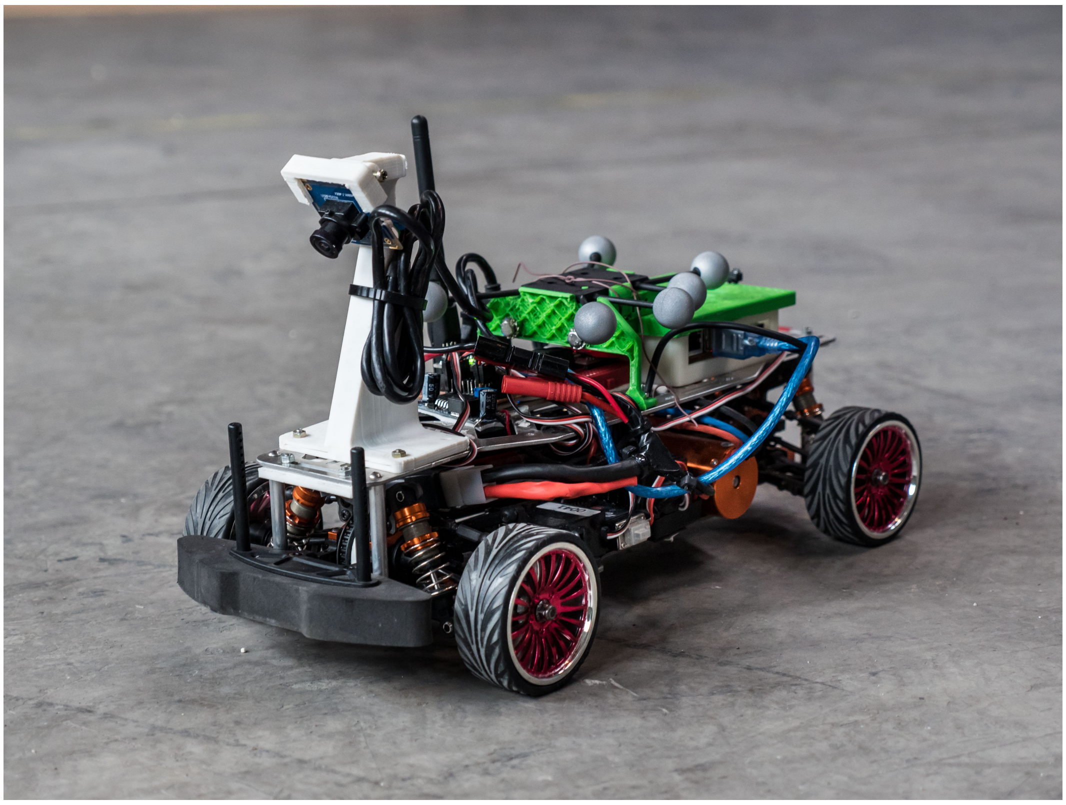


**Figure 2-2:** Picture of the Berkeley Autonomous Race Car (BARC) system

## 2-2-1 Inertial Measurement Unit

The BARC system is kitted with a 9 degrees of freedom IMU. One single System on a Chip (SoC) delivers 3 degrees of rotational velocity $(\omega_x, \omega_y, \omega_z)$, 3 degrees of planar acceleration $(a_x, a_y, a_z)$ and the magnetic field strength in 3 directions $(B_x, B_y, B_z)$. These quantities are measured in a fixed right-hand coordinate system. The IMU is mounted backwards on the vehicle, resulting in the quantities belonging to the x- and y-axis having their sign inverted when compared to the vehicle axis system. This can also be seen in Figure 2-3.

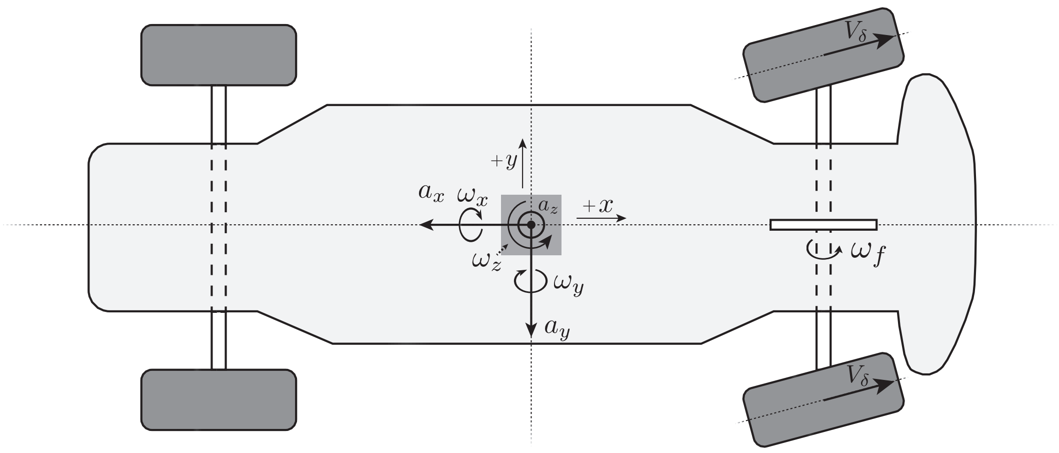


**Figure 2-3:** Top view of the BARC and sensors; the IMU (grey box) together with the coordinate system it uses and the encoder (white disc) and the obtained $V_\delta$ from the encoder. Note that $a_z$ is pointing upwards.

The IMU is mounted approximately 3 cm above the assumed centre of gravity of the vehicle. It is possible to transform the measured quantities to the correct centre of gravity quantities by a coordinate transformation. However, for the remainder of this project it is assumed that the IMU measures the centre of gravity acceleration and rotational velocity, since the rolling and pitching motion of the vehicle is not that large ($[-5°, 5°]$). The conversion from IMU states to vehicle centre of gravity states is therefore defined as:

$$a_{x,\text{vehicle}} = -a_{x,\text{meas}} \qquad \omega_{x,\text{vehicle}} = -\omega_{x,\text{meas}} \tag{2-1}$$

$$a_{y,\text{vehicle}} = -a_{y,\text{meas}} \qquad \omega_{y,\text{vehicle}} = -\omega_{x,\text{meas}} \tag{2-2}$$

$$a_{z,\text{vehicle}} = a_{z,\text{meas}} \qquad \omega_{z,\text{vehicle}} = \omega_{z,\text{meas}} \tag{2-3}$$

### 2-2-2 Encoders

The second sensor of the BARC that can be used for this project is its wheel speed encoder. A plastic disc with four (almost) evenly spaced magnets is connected to each axle. Next to the disc is a Hall sensor that picks up a change in magnetic field and toggles a discrete signal on the output pin. The setup is shown in Figure 2-4.

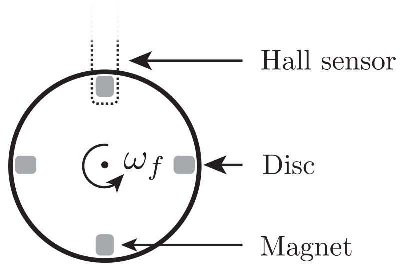


**Figure 2-4:** The encoder wheel that is connected to the front and read axle of the BARC, together with the magnets and Hall sensor.

The microcontroller attached to the sensor will measure the time between two changes in the discrete signal. Unfortunately, the implementation used during experiments does not differentiate between a rising and a falling edge of the signal, so the last two intervals are needed to obtain the total time it takes for one quarter rotation of the disc. The speed the encoder measures can then be calculated using the following relationship:

$$V_{enc,i} \approx \frac{\Delta s}{\Delta t} = \frac{2\pi R_w}{N(t_k - t_{k-2})} \quad i = \{f, r\} \tag{2-4}$$

Where $\Delta s$ is the displacement of one quarter rotation and $\Delta t$ its respective duration, $R_w$ represents the wheel diameter, $N$ the amount of magnets changes per rotation and $t_k - t_{k-2}$ the time of one quarter rotation, namely two changes in signal. The front axle of the BARC is not powered and does not feature any form of a differential. Therefore both front wheels turn at the same rotational speed. The velocity the encoder measures is therefore in fact $V_\delta$, the speed of the wheels rotating in the direction of the wheel centre line, which is offset by $\delta_f$ from the vehicle center line. The speed $V_\delta$ can be calculated by projecting the front axle speed $V_f$ onto the center line of the wheel. $V_f$ is composed by the longitudinal velocity $V_x$ and the corrected front axle lateral velocity $V_{yf}$:

$$V_f = \sqrt{\left(V_x^2 + V_{yf}^2\right)} \qquad V_{yf} = V_y - l_f\omega_z \tag{2-5}$$

$$V_\delta = V_f \cos\alpha_f = \sqrt{(V_x^2 + (V_{yf} - l_f\omega_z)^2)}\cos(\delta_f - \tan^{-1}\left(\frac{V_{yf}}{V_x}\right)) \tag{2-6}$$

### 2-2-3 Camera

The BARC features a 2 megapixel camera that can record image sequences at different resolutions and frame rates. In the next chapter the details of the CVA will be discussed in more detail, but for now the position of the CVA in relation to an observer will be discussed. The camera of the BARC is tilted down at roughly 20°. The biggest part of the image consequently represents the ground plane. The CVA will be able to follow one or more parts of that ground plane and estimate the velocity of those parts in the longitudinal and lateral direction of the vehicle. The velocity of the centre of gravity of the vehicle is sometimes also referenced as the egomotion. It can be derived by assuming not the tracked parts in the image are moving, but the vehicle itself is moving. This is an assumption that for now will be assumed valid, but it is an interesting consideration for later research. The egomotion of the vehicle can be retrieved using the kinematic relationship:

$$V_{x,i}(k) = -\left(\frac{x_i(k) - x_i(k-1)}{\Delta t} + \omega_z(k) y_i(k-1)\right) \tag{2-7}$$

$$V_{y,i}(k) = -\left(\frac{y_i(k) - y_i(k-1)}{\Delta t} - \omega_z(k) x_i(k-1)\right) \tag{2-8}$$

Where $x_i(k), y_i(k)$ are the real world coordinates of some point or image patch (a group of pixels). In this case it is point number $i$ out of $N_p$ total points in frame $F_k$ which is sample number $k$ in a series of samples. The pair $V_{x,i}(k), V_{y,i}(k)$ is the egomotion of the vehicle according to that point or patch in the image tracked from frame $k-1$ to $k$. For each tracked point or patch the egomotion can be calculated. In the case of $N_p$ points, this will yield a measurement vector of:

$$z(k) = \begin{bmatrix} V_{x,1} \\ \vdots \\ V_{x,N_p} \\ V_{y,1} \\ \vdots \\ V_{y,N_p} \end{bmatrix} \tag{2-9}$$

To calculate the real world position of points based on their pixel location, an estimate of the current pose of the camera is needed. This pose can be calibrated before operation of the vehicle, but due to suspension movement of the vehicle the camera pose changes during operation. Therefore the CVA depends on an estimate of the pitch angle $\theta$ and roll angle $\phi$ of the vehicle. The decision was made to implement an observer for the roll angle to test how effective running an online observer for this purpose is. The roll angle had by far the greatest influence on removing errors from the CVA and implementing an observer for this is more feasible than implementing a pitch angle observer. The roll angle observer will be discussed in Section 2-5.

## 2-3 Velocity estimation

The main component of the observation algorithm is the velocity observer. It combines all data from the sensors together with a hybrid dynamic-kinematic model of the system to yield an estimate of the longitudinal and lateral velocity. These two can then be combined to estimate the sideslip angle.

### 2-3-1 Process model

For the observer in this thesis the kinematic model based observer from [33] was chosen as a basis. The kinematic observer still has the promising features of not requiring an on- or offline system identification, being very good during highly dynamic maneuvers and providing decent estimates on steady-state maneuvers when amended with some smart functions or heuristics [11, 34]. The calculated output from the system model will firstly be the front wheel speed $V_\delta$. Later the output of the system will be compared to the measured front wheel speed. This gives the following system as a starting point:

$$\begin{bmatrix} \dot{V}_x \\ \dot{V}_y \end{bmatrix} = \begin{bmatrix} 0 & \omega(t) \\ -\omega(t) & 0 \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \end{bmatrix} \tag{2-10}$$

$$y_1 = h_1(\mathbf{x}) = \Big[V_\delta(V_x, V_y)\Big] = \left[\sqrt{(V_x^2 + (V_{yf} - l_f\omega_z)^2)}\cos\Big(\delta_f - \tan^{-1}\Big(\tfrac{V_{yf}}{V_x}\Big)\Big)\right] \tag{2-11}$$

Where $\mathbf{x}$ is the state vector $[V_x, V_y]$. To combat the main weaknesses of the kinematic observer, namely the loss of observability during straight driving and the sensitivity to sensor drift, the calculated lateral acceleration is used as an output that is later compared with the actual measured lateral acceleration. This is based on the method proposed by [29]. The lateral acceleration is calculated under a zero yaw acceleration assumption together with a tire model based on the hyperbolic tangent function, adopted from [11]. This creates a second nonlinear output function $h_2(\mathbf{x})$:

$$y_2 = h_2(\mathbf{x}) = \frac{1}{m}\left(F_{yf}\cos\delta_f + F_{yr}\right) = \hat{a}_y \tag{2-12}$$

$$F_{yi} = \frac{C_i}{k_i}\tanh k_i\alpha_i \quad i = \{f, r\} \tag{2-13}$$

$$\alpha_f = \left(\delta_f - \frac{V_y + l_f\omega_z}{V_x}\right) \quad \alpha_r = \left(-\frac{V_y - l_r\omega_z}{V_x}\right) \tag{2-14}$$

The measurements from the camera can also be added to the hybrid observer, to correct the observer estimate when the calculated lateral acceleration is no longer valid. This could happen when the tire model from (2-13) is not up to date or when the yaw acceleration is large and (2-12) is no longer a valid equation. As was already shown in [14], the measurements from the camera can be added to the measurement vector of the kinematic model based observer. The difference to [14] will be that no median filtering is applied to the velocities coming from all the tracked points resulting in one median speed, but all measured velocities will be entered into the measurement vector as separate entries. To compare the egomotion according to the camera with the estimated velocities from the process model a vector $y_3 \in \mathbb{R}^{2N_P \times 1}$ needs to be added to the process model, with $N_p$ representing the number of tracked points or patches on the ground plane:

$$y_3 = h_3(\mathbf{x}) = \begin{bmatrix} V_x \\ \vdots \\ V_x \\ V_y \\ \vdots \\ V_y \end{bmatrix} \tag{2-15}$$

The hybrid kinematic model with the three outputs $[y_1, y_2, y_3]^T$ will form the basis for the observer described in the next section.

## 2-3-2 Discretization

The process model is now defined in state space form as a Linear Parameter Varying system with a nonlinear output function $y = h(\mathbf{x})$. The observer will later be implemented as a digital and discrete algorithm running at 30 Hz. This is the highest frame rate at which the camera can operate. Before a discrete observer can be discussed, the process model needs to be discretized first. The parameter varying state matrix $A(t)$ and constant input matrix $B$ can be discretized into respectively $\Phi_k$ and $\Gamma_k$:

$$\Phi_k = e^A T_s = \begin{bmatrix} cos(\omega_{z,k}T_s) & \sin(\omega_{z,k}T_s) \\ -\sin(\omega_{z,k}T_s) & cos(\omega_{z,k}T_s) \end{bmatrix} \tag{2-16}$$

$$\Gamma_k = \int_0^{T_s} e^{As} ds \; B = \frac{1}{\omega_{z,k}} \begin{bmatrix} \sin(\omega_{z,k}T_s) & 1 - cos(\omega_{z,k}T_s) \\ cos(\omega_{z,k}T_s) - 1 & \sin(\omega_{z,k}T_s) \end{bmatrix} \tag{2-17}$$

Where $T_s$ represents the sample time and $\omega_{z,k}$ the yaw rate at sample $k$. The subscript $\ldots_k$ will be used from now on to indicate sample number $k$ taken from a continuous time signal at interval time $T_s$. The full discrete process model that forms the basis for the observer can now be presented.

### 2-3-3 Observer structure

Next to the discrete state space matrices $\Phi_k, \Gamma_k$, the state vector $\mathbf{x}_k$ and the input vector $\mathbf{u}_k$, noise on the state update and the measurements can be included in the form of the vectors $\mathbf{w}_k$ and $\mathbf{v}_k$. In discrete form, the system can now be written as:

$$\mathbf{x}_k = \Phi_k \mathbf{x}_{k-1} + \Gamma \mathbf{u}_k + \mathbf{w}_k \tag{2-18}$$

$$\mathbf{y}_k = h_k(\mathbf{x}_k) + \mathbf{v}_k \tag{2-19}$$

The noise vectors $\mathbf{w}_k$ and $\mathbf{v}_k$ are supposed to be random processes. The process noise vector represents the error in the state dynamics coming from unmodelled effects as well as the noise on the IMU signals used in the kinematic equations. The measurement noise vector represents the error in measurements that cannot be predicted using the output function and can be attributed to noise or unmodelled dynamics. The covariance matrix for the noise vectors can be defined as:

$$E \begin{bmatrix} \mathbf{w}(k) \\ \mathbf{v}(k) \end{bmatrix} \begin{bmatrix} \mathbf{w}(j)^T & \mathbf{v}(j)^T \end{bmatrix} = \begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \Delta(k-j) \tag{2-20}$$

The $\Delta(x)$ function is zero for all $x$, except $x = 0$. Both noise vectors are assumed to contain zero-mean, Gaussian noise. Consequently no cross-correlation between the two vectors is present and cross-correlation matrix $S$ is assumed zero. The knowledge of the $Q$ and $R$ matrix can later be used to create a better observer. The aim of the observer is to correct the predictions of the process model with a set of measurements. The vector of measurements $\mathbf{z}_k$ used for this in the observer is constructed as:

$$\mathbf{z}_k = \begin{bmatrix} V_{enc,F} \\ a_{y,meas} \\ V_{x,1} \\ \vdots \\ V_{x,N_p} \\ V_{y,1} \\ \vdots \\ V_{y,N_p} \end{bmatrix} \tag{2-21}$$

Now a suitable observer type must be selected that can deal with the process and measurement noise. The IMU measurements are one of the key elements of the process model. It is known that the planar acceleration signals coming from the IMU are particularly noisy. To counter these noisy signals the Kalman Filter was selected as the basis for the observer in this project. Its properties of dealing in a powerful way with noise on the process and the measurements were a good starting point. The chosen outputs of the process model make the output function $h(\mathbf{x})$ nonlinear. To still be able to create an observer that uses (most of) the benefits of the Linear Kalman Filter, but allows for nonlinear output functions, an Extended Kalman Filter (EKF) was chosen.

In a nonlinear system the $A$ and $C$ state space matrices are replaced with the nonlinear state transition functions $f(\mathbf{x})$ and output function $h(\mathbf{x})$. The EKF mitigates the problem of a nonlinear state transition or output function by replacing the $A$ and $C$ matrix in the Discrete time Algebraic Ricatti Equation (DARE) with the Jacobian of their nonlinear equivalent function function $f(\mathbf{x})$ and $h(\mathbf{x})$. The EKF is not an optimal observer in terms of yielding a minimum covariance zero mean estimate, since it makes these assumptions. (2-18) does not feature a function $f(\mathbf{x})$, since the state transition is defined using a Linear Parameter Varying $A$ matrix which is independent of $\mathbf{x}_k$. Therefore only the Jacobian of the output function $H_k(\mathbf{x})$ will be used in the observer. The definition of the Jacobian of $h(\mathbf{x})$ can be seen in (2-24), for the Jacobian of the used output vector see Appendix B-1. The EKF starts at every time step with a prediction of the new states and state covariance based on previous information. These estimates are called the *a priori* estimates:

$$\hat{x}_{k|k-1} = \Phi_k \hat{x}_{k-1} + \Gamma_k u_k \tag{2-22}$$

$$P_{k|k-1} = \Phi_k P_{k-1|k-1} \Phi_k^T + Q_k \tag{2-23}$$

In this set of equations a hat indicates an estimated quantity and the subscript $_{k|k-1}$ represents a calculation at time step $k$ based on the information of time step $k-1$ . The estimated state covariance matrix is indicated with $P_k$ and $Q_k$ is the covariance matrix of the process noise $\mathbf{w}_k$. After the prediction step, a correction is made based on the measurements and the Kalman filter gain $K_k$, which is recalculated every time step. This is necessary because of the time dependent $\Phi_k$ matrix and output function $h_k(\mathbf{x}_k)$. By applying the correction the *a posteriori* estimates are obtained:

$$H_k = \left.\frac{\partial h}{\partial \mathbf{x}}\right|_{x=\hat{x}_{k|k-1}} \tag{2-24}$$

$$K_k = P_{k|k-1} H_k^T \left( H_k P_{k|k-1} H_k^T + R_k \right)^{-1} \tag{2-25}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k \left( \mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1}) \right) \tag{2-26}$$

$$P_{k|k} = \left( I - K_k H_k \right) P_{k|k-1} \tag{2-27}$$

In these equations $R_k$ represents the covariance matrix of the measurement noise vector $\mathbf{v}_k$. The a posteriori estimate $\hat{\mathbf{x}}_{k|k}$ is the vector that can later be used by a control system to perform control actions.

## 2-4 Dynamic Covariance Estimation

Normally, the $Q$ and $R$ matrix contain static values, since they represent unmodelled dynamics and sensor noise, which is assumed to be a static quantity. In Chapter 4 it will be shown that applying a static $Q$ and $R$ matrix is not sufficient for integrating camera measurements into the hybrid model based observer.

The covariance submatrix of the camera measurement noise will be adjusted based on functions that indicate if a certain point is more or less prone to error. By doing this, the chosen observer becomes an adaptive observer. It can put more or less weight on certain signals depending on certain indications from the system states or the tracking parameters by adjusting the variance of that measurement. The DCE calculates these covariance matrices based on the yaw rate and the point locations. Before the exact methodology can be discussed, more knowledge of the CVA is needed, since a large part of the DCE is based on the properties and performance of the CVA.

## 2-5 Roll angle observer

Next to the velocity observer, a roll angle observer was designed. As said before, this was done as a proof of concept for the online estimation of the camera pose to correct the coordinate transformation from pixel location to real world location. The full concept of this transformation is explained in Section 3-1. In this section, the methodology for estimation of the roll angle itself is discussed.

To estimate the roll angle $\phi$, a second order torsional mass-damper-spring model excited by lateral acceleration was adopted. The sprung mass of the vehicle is suspended above a torsional mass and -damper. Based on the amount of lateral acceleration the mass will rotate. When the mass is suspended at a sufficiently small angle $\phi$, the roll acceleration $\ddot{\phi}$ can be written as:

$$I_{xx}\ddot{\phi} = -K_{\phi}\phi - b_{\phi}\dot{\phi} + m_s g h_s \phi + m_s h_s a_y \tag{2-28}$$

The model features the following parameters, which can also be seen in Figure 2-5.:

- $h_s$: The distance from the centre of mass to the point it rotates around (roll centre).
- $I_{xx}$: The moment of inertia of the rotating system around the $x$ axis
- $K_{\phi}$ & $b_{\phi}$: Respective rotational stiffness and damping.
- $m_s$: sprung mass of the vehicle, i.e. all components that are mounted on the suspension. Approximated to be $m_s = 1.7\,\mathrm{kg}$
- $g$: gravitational acceleration, $g = 9.81\,\mathrm{m\,s^{-2}}$.

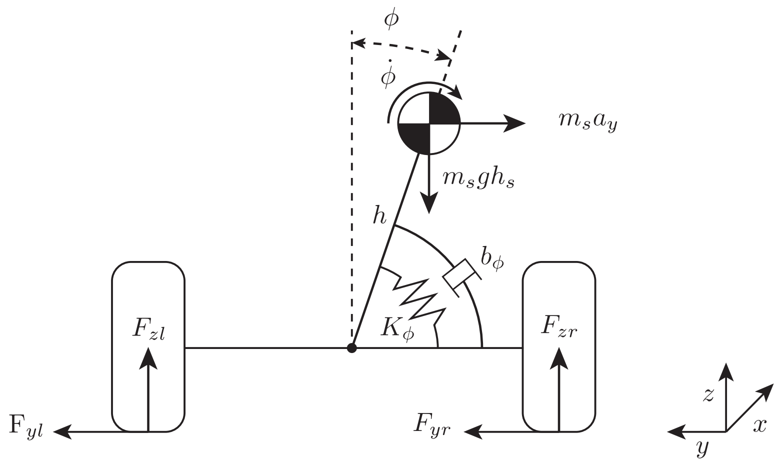


**Figure 2-5:** Schematic view of the sprung vehicle mass rolling during a left hand corner. Image adapted from [48]

The road disturbance forces $F_{z,i},\ i = \{l, r\}$ coming from the tires are not modelled and seen as an external disturbance. Since the tires are solid rubber and not inflated, damping and stiffness in the tires between the ground and the suspension were neglected. Finally it was also assumed that the vehicle is always driving on a flat surface and therefore road banking is not accounted for. Since the roll observer was implemented just as a proof of concept a simple model was used as a starting point and these assumptions were seen as acceptable. The second order differential equation can be put in state space form so it can later be used for creating an observer. The roll velocity is chosen as the single output, since it is a quantity the IMU can observe:

$$\begin{bmatrix} \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \frac{1}{I_{xx}} \begin{bmatrix} 0 & 1 \\ -K_\phi + m_s g h & -b_\phi \end{bmatrix} \begin{bmatrix} \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{m_s h}{I_{xx}} \end{bmatrix} a_y \tag{2-29}$$

$$y = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \phi \\ \dot{\phi} \end{bmatrix} \tag{2-30}$$

To improve the estimate of the roll angle, the system was put into an observer. Here, the Linear Kalman Filter was used, since the system is a Linear Time Invariant system. The roll angle observer uses the measured lateral acceleration as an input and computes the roll angle and roll velocity based on that using the discretized version of (2-29). The calculated roll velocity is then compared with the measurement from the IMU. Discretization was done in the same way as in (2-16) to (2-17). The full roll observer is defined as:

Process model:

$$\mathbf{x}_k = \Phi\mathbf{x}_{k-1} + \Gamma\mathbf{u}_k + \mathbf{w}_k \tag{2-31}$$

$$\mathbf{y}_k = C\mathbf{x}_k + \mathbf{v}_k \tag{2-32}$$

$$\mathbf{z}_k = \left[\dot{\phi}_{meas}\right] \tag{2-33}$$

Prediction:

$$\hat{\mathbf{x}}_{k|k-1}- = \Phi\hat{\mathbf{x}}_{k-1} + \Gamma u_k \tag{2-34}$$

$$P_{k|k-1} = \Phi P_{k-1|k-1}\Phi^T + Q \tag{2-35}$$

Correction:

$$K_k = P_{k|k-1}C^T\left(CP_{k|k-1}C^T + R\right)^{-1} \tag{2-36}$$

$$\hat{\mathbf{x}}_{k|k} = \mathbf{x}_{k|k-1} + K_k\left(z_k - C\mathbf{x}_{k|k-1}\right) \tag{2-37}$$

$$P_{k|k} = (I - K_kC)\,P_{k|k-1} \tag{2-38}$$

$$\tag{2-39}$$

The unknown system parameters $K_\phi, b_\phi, h_s, I_xx$ and covariance matrices $Q, R$ are respectively identified and optimized in Section 5-4. The roll angle estimation will be run as a separate observer next to the main EKF that is computing the vehicle velocity.

## 2-6 Observer summary

A full overview of the observer structure can be seen in Figure 2-6. Figure 2-1 has been amended with labels for all signals, the process model, measurement model and EKF equations all have been added. It can be seen that the process model is only fed by IMU signals and the measurement model requires IMU, encoder and camera signals. The CVA and DCE blocks have been drawn in, but their exact workings will be discussed in the next chapters.

## 2-7 Conclusion

The chosen methodology for combining the data from the IMU, wheel speed encoder and the camera has been presented in this chapter. An observer combining the kinematic model with a small dynamic model forms the basis. The measurements coming from the camera of the vehicle are then added to correct the observer when assumptions of the dynamical model are no longer valid. This is complemented by a measurement covariance matrix that is automatically adjusted during operation of the filter to better process the camera measurements, the so called Dynamic Covariance Estimation. An auxiliary observer runs in parallel to the first observer to estimate the body roll angle. This is done to allow for better velocity estimation using the CVA.

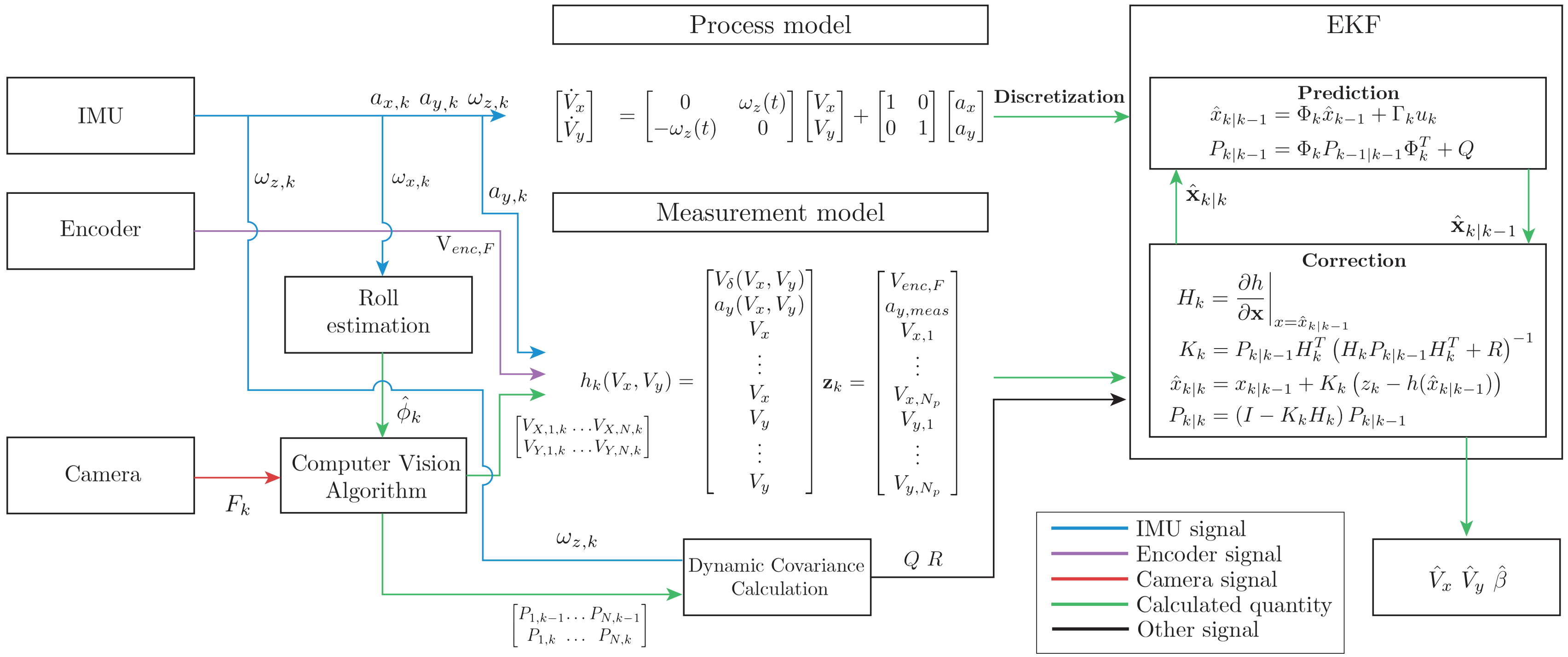

IMU
Encoder
Camera
Roll estimation
Computer Vision Algorithm
Dynamic Covariance Calculation
$a_{x,k}$ $a_{y,k}$ $\omega_{z,k}$
$\omega_{z,k}$
$\omega_{x,k}$
$a_{y,k}$
$V_{enc,F}$
$\hat{\phi}_k$
$F_k$
$\begin{bmatrix} V_{X,1,k} \dots V_{X,N,k} \\ V_{Y,1,k} \dots V_{Y,N,k} \end{bmatrix}$
$\omega_{z,k}$
$\begin{bmatrix} P_{1,k-1} \dots P_{N,k-1} \\ P_{1,k} \dots P_{N,k} \end{bmatrix}$
$Q$ $R$
Process model
$\begin{bmatrix} \dot{V}_x \\ \dot{V}_y \end{bmatrix} = \begin{bmatrix} 0 & \omega_z(t) \\ -\omega_z(t) & 0 \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \end{bmatrix}$
Discretization
Measurement model
$h_k(V_x, V_y) = \begin{bmatrix} V_\delta(V_x, V_y) \\ a_y(V_x, V_y) \\ V_x \\ \vdots \\ V_x \\ V_y \\ \vdots \\ V_y \end{bmatrix}$
$\mathbf{z}_k = \begin{bmatrix} V_{enc,F} \\ a_{y,meas} \\ V_{x,1} \\ \vdots \\ V_{x,N_p} \\ V_{y,1} \\ \vdots \\ V_{y,N_p} \end{bmatrix}$
EKF
Prediction
$\hat{x}_{k|k-1} = \Phi_k \hat{x}_{k-1} + \Gamma_k u_k$
$P_{k|k-1} = \Phi_k P_{k-1|k-1} \Phi_k^T + Q$
$\hat{\mathbf{x}}_{k|k}$
$\hat{\mathbf{x}}_{k|k-1}$
Correction
$H_k = \left.\frac{\partial h}{\partial \mathbf{x}}\right|_{x=\hat{x}_{k|k-1}}$
$K_k = P_{k|k-1} H_k^T \left(H_k P_{k|k-1} H_k^T + R\right)^{-1}$
$\hat{x}_{k|k} = x_{k|k-1} + K_k \left(z_k - h(\hat{x}_{k|k-1})\right)$
$P_{k|k} = (I - K_k H_k) P_{k|k-1}$
$\hat{V}_x$ $\hat{V}_y$ $\hat{\beta}$
IMU signal
Encoder signal
Camera signal
Calculated quantity
Other signal

**Figure 2-6:** Overview of the hybrid kinematic observer together with the sensors feeding the observer, process and measurement model and the to be discussed Computer Vision Algorithm and Dynamic Covariance Estimation

Chapter 3

# Computer Vision Measurements

The next step in obtaining a robust observer for the body sideslip angle is creating an algorithm that can deliver measurements on the velocity of the vehicle, based on the pictures the camera captures. The camera used for the Computer Vision Algorithm (CVA) is a 2 Megapixel camera with a 1/3 inch CMOS sensor and a 2.1mm focal distance super wide angle lens. In the final observer, the camera should provide a vector of measures containing the egomotion for every tracked point or patch on the ground plane. In this chapter the camera model converting world points to pixels will be discussed first, then the image preprocessing will be discussed, succeeded by the feature detection, tracking and coordinate transformation. Finally an overview of the algorithm in pseudo code will be presented and the sensitivity of some parameters of the algorithm is shown.

## 3-1 Camera placement and modelling

The overall goal of the CVA is to estimate the speed of the vehicle based on images from the camera. As said before, the camera is pointed downwards at roughly 20°. A result of this is that the camera mostly sees the ground plane. The first step in devising the algorithm is to understand how points on the ground plane in the real world end up as pixels in an image. To do this, a camera model is adopted.

### 3-1-1 Intrinsic parameters

Several models for modelling a camera exist, such as the perspective (sometimes called pinhole), affine, orthographic and weak perspective model. The perspective model is widely used in camera calibration techniques and also used in this project. The assumption of the model is that all light rays captured by the camera converge to one point, camera's optical center, in a straight line. To illustrate this in mathematics, consider a point P in a real world coordinate system X-Y-Z. The world's X-Y plane is parallel with the camera's x-y plane and the origin

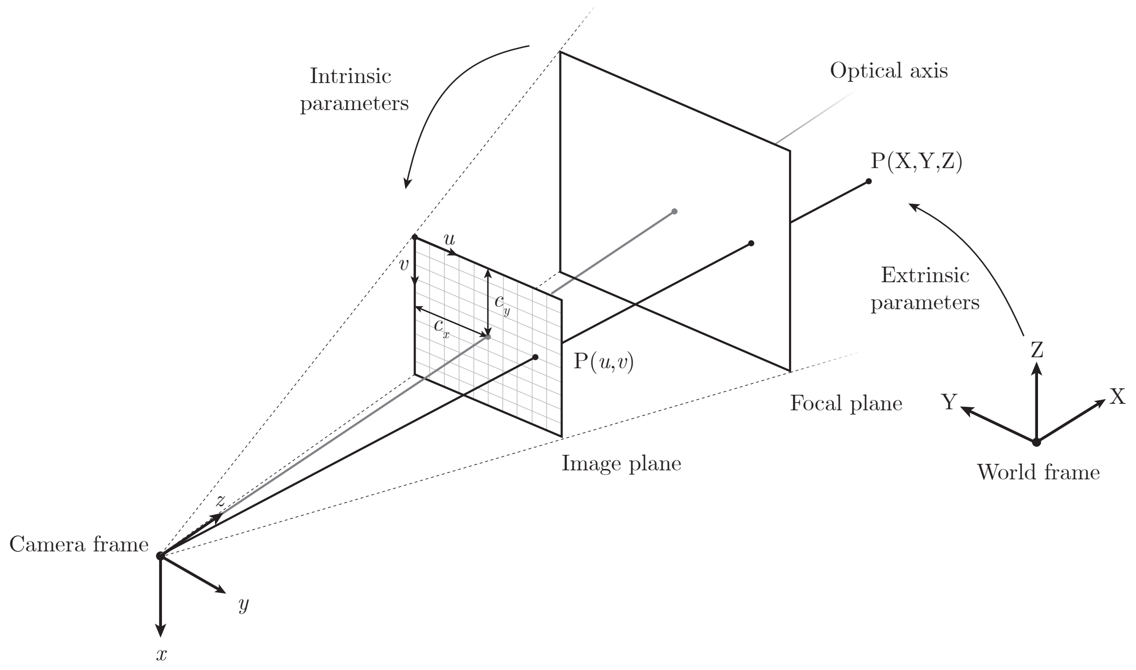


**Figure 3-1:** Overview of the transformation from world point to focal plane and then to image plane by transforming with first with extrinsic and then intrinsic parameters.

of the world's X-Y plane lies on the camera's z-axis. This is also drawn in Figure 3-1. A world point $P(X,Y,Z)$ will be projected on the camera's image sensor at $p(x,y)$:

$$\frac{Z}{f_x} = \frac{X}{x} \quad \frac{Z}{f_y} = \frac{Y}{y} \tag{3-1}$$

Where $f_i$ $i = \{x,y\}$ is the focal length of the camera along the respective axes, $X,Y$ are world coordinates and $x,y$ are image frame coordinates. Usually the focal lengths have roughly the same value, but due to lens inaccuracies, sensor flaws or digital stretching small variations can occur. In an image frame, the respective pixels are not described in a normal x-y coordinate system, but in the $u$-$v$ coordinate system. This system has its origin in the top left corner of an image, $u$ is pointing right side positive and $v$ points downward positive. The pixel coordinates can be written as:

$$u = x + c_x \; v = -y + c_y \tag{3-2}$$

Where $c_x, c_y$ is the location of the optical centre when regarded from the top left corner of the image. The image coordinate $p(u,v)$ (a small letter $p$ will indicate an image point, a capital $P$ will indicate a world point) can be written as a matrix multiplication of the camera parameters and a world point as long as the coordinate frame of the world point is correctly aligned with the camera reference frame:

**Table 3-1:** Estimated values for radial distortion and intrinsic camera parameters

| **Distortion** | | **Intrinsic parameters** | |
|---|---|---|---|
| Parameter | Value | Parameter | Value |
| $k_1$ | -0.3124 | $f_x$ | 613.2 |
| $k_2$ | 0.0782 | $f_y$ | 621.4 |
| | | $c_x$ | 528.2 |
| | | $c_y$ | 369.8 |

$$\lambda \underbrace{\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}}_{p} = \underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathcal{K}} \underbrace{\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}}_{P} \tag{3-3}$$

Where $\mathcal{K}$ is the intrinsic parameter matrix and $\lambda$ is a scaling parameter. After multiplication the $u$ and $v$ coordinate can be divided by $\lambda$ to obtain the true pixel location. The four intrinsic parameters can be identified using the `calibrateCamera` function of OpenCV [49]. It is based on the method of [50], where a set of images from a checkerboard with known checker size (in this project 7.8cm) is used to estimate the parameters. The results of this calibration can be seen in Table 3-1. This matrix multiplication assumes the coordinate system of the world is aligned with the one of the camera. Usually this is not the case. To align these two systems, the extrinsic parameters are also needed which are discussed in Subsection 3-1-3.

### 3-1-2 Distortion removal

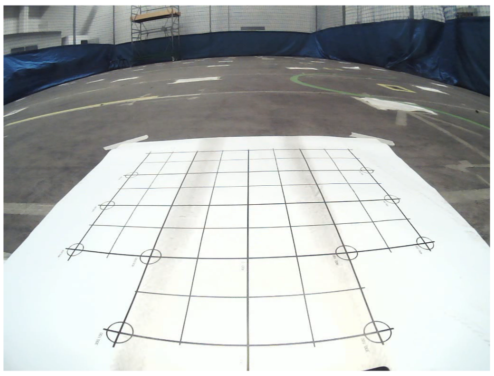
For now it has been assumed that all light rays travel in a straight line from the world point to the camera's optical center. Camera lenses are, however, curved and therefore bend light differently at different places of the lens. Normally straight lines will therefore appear curved in the image. An example of this phenomenon can be seen in Figure 3-2. In the image radial distortion is visible. Other types of distortion, such as tangential and prism distortion also exist [51]. Fortunately, the distortion of the light due to the lens properties can be removed after the image is captured. The aforementioned `calibrateCamera` function can also identify distortion parameters. It was found that the used camera only exhibits radial distortion. The radial distortion can be modelled with a fourth order equation:

$$u_{\text{dist}} = u_{\text{cor}}(1 + k_1 r^2 + k_2 r^4) \tag{3-4}$$

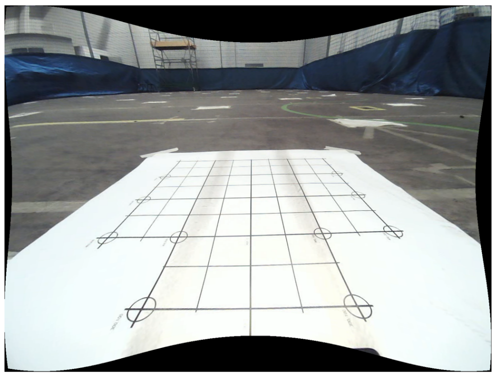
$$v_{\text{dist}} = y_{\text{cor}}(1 + k_1 r^2 + k_2 r^4) \tag{3-5}$$

$$\text{s.t. } r^2 = (u_{\text{dist}} - c_x)^2 + (v_{\text{dist}} - c_y)^2 \tag{3-6}$$

Where $c_x, c_y$ is again the optical center, $k_1, k_2$ are the unknown distortion parameters, $u_{\text{dist}}, v_{\text{dist}}$ is the distorted image point and $u_{\text{cor}}, v_{\text{cor}}$ is the original point if all light rays would travel straight as in (3-3). The distortion parameters were identified with the `calibrateCamera` function of OpenCV and results are shown in Table 3-1. After calibrating these parameters, distortion can be removed from the image as is shown in Figure 3-2b.



**(a)** Raw image captured from BARC camera. Notice the radial distortion at the normally straight grid



**(b)** Undistorted image from BARC Camera. Notice the black areas of the image that are lost due to undistortion

**Figure 3-2:** Raw camera image with distortion and the same image after distortion removal using calibrated distortion parameters

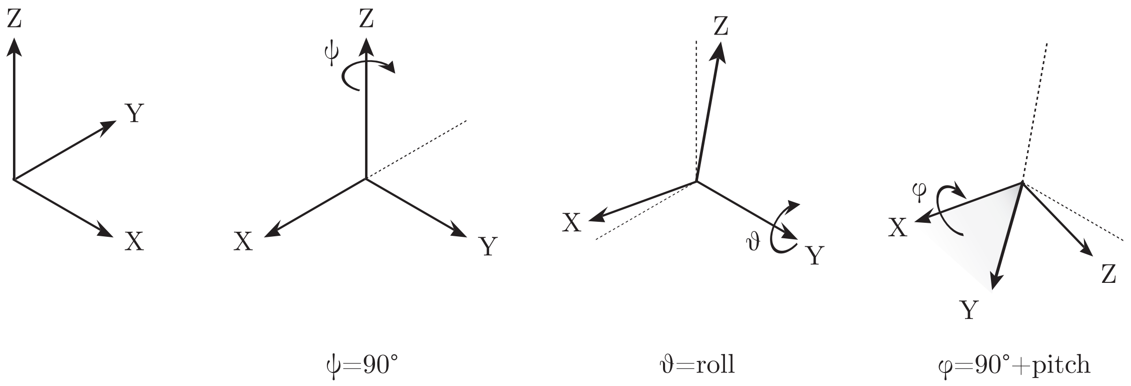


**Figure 3-3:** Rotation sequence for going from a vehicle coordinate system, X = longitudinal, Y = lateral, Z= vertical, to a camera coordinate system, including the places to introduce the additional roll and pitch angle of the camera during dynamic maneuvers.

### 3-1-3 Extrinsic parameters

So far, it has been assumed that the coordinate system of the focal plane and the world coordinate system had parallel $X$-$Y$ planes. However, with the camera mounted on top of the car and angled down, the ground plane and the image plane of the camera are not parallel. This can be solved by calibrating a coordinate transformation from the original world reference frame to a frame where the $X$-$Y$ plane is aligned with the camera frame. The world reference frame is chosen to be the same as was used in Chapter 1 when discussing vehicle dynamics. That is one where the origin of the $X$-$Y$ ground plane is located below the BARC's (estimated) centre of gravity, the $X$-axis points forward (positive longitudinal direction), the $Y$-axis left (positive lateral direction) and the $Z$-axis points upwards. This can be seen in the first image of Figure 3-3.

This coordinate transformation can be done by multiplying the original coordinate with a rotation matrix and adding a translation vector. The rotation matrix and translation vector combined are also referenced as the extrinsic parameters. As the final pose of the camera is the result of several rotations of different angles around different axes, so is the final rotation matrix. The final rotation matrix can be written as the product of several square $3 \times 3$ orthonormal matrices. The matrices for a rotation around respectively the X-, Y- and Z-axis in a right hand system are defined as:

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \; R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \; R_z(\omega) = \begin{bmatrix} \cos\omega & -\sin\omega & 0 \\ \sin\omega & \cos\omega & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3-7}$$

During driving, the pose of the camera may vary slightly from the pose when the vehicle is not moving. Because the extrinsic parameter are calibrated when the vehicle is standing still, the extra roll or pitch of the camera during driving will cause errors in the speed calculation later on. By selecting a useful sequence of axes to rotate around, the (estimated) extra roll and pitch angle of the vehicle during driving can later be added as arguments to the multiplication sequence to compensate for the extra movement. When decomposing the final rotation in three steps and selecting three different axes, there still exist at least six different ways to arrive to the same rotation (X-Y-Z, X-Z-Y, Y-Z-X, etc.). Here a sequence consisting of a rotation $\psi_0$ around the Z-axis, then a rotation $\theta_0$ around the Y-axis and finally a rotation $\phi_0$ around the X-axis is chosen, see also Figure 3-3. The rotation matrix can then be formulated as:

$$R = R_x(\phi_0 + \phi_k) \; R_y(\theta_0 + \theta_k) \; R_z(\psi_0) \tag{3-8}$$

The pose of the camera when the vehicle is standing still and not experiencing loads is defined as $(\psi_0, \theta_0, \phi_0)$. The extra pitch and rotation of the camera at sample $k$ are $\theta_k$ and $\phi_k$. Defining the pose of the camera in this way has the advantage that during driving the extra movement of the camera can be removed by estimating the body roll and pitch angle.

The extrinsic parameters can be calculated by measuring the rotation angles of the camera by hand and estimating the translation. However, this will be very inaccurate. Therefore another way of estimating the extrinsic parameters was used, namely decomposing the homography matrix $\mathcal{H}$. Combining the intrinsic parameter matrix and the extrinsic parameter matrix, the full transformation from unaligned world point to pixel can be written out. To enable this transformation, the world point is amended with a fourth coordinate in the form of a 1. In this form, the coordinate is called a homogeneous coordinate:

$$\lambda \underbrace{\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}}_{p} = \underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathcal{K}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathcal{S}} \underbrace{\left[\begin{array}{c|c} R & t \\ \hline 0 & 1 \end{array}\right]}_{\mathcal{E}} \underbrace{\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}_{P} \tag{3-9}$$

$$\lambda \, p = \mathcal{K}\mathcal{S}\mathcal{E}P \tag{3-10}$$

Where $\lambda$ is again a scaling parameter, $\mathcal{S}$ is a selection matrix and $\mathcal{E}$ is the extrinsic parameter matrix. The values of the intrinsic parameter matrix were already determined and are the same for every experiment, since they only depend on the physical properties of the camera. Because of the distortion parameters being known, an undistorted image such as Figure 3-2b can be used as a basis to estimate the extrinsic parameters. By selecting 4 or more points in the image with a known physical location, the homography matrix $\mathcal{H}$ can be estimated. The homography matrix transforms points from world coordinates to pixel coordinates, by assuming that all selected points have zero height, i.e. $Z = 0$:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathcal{H} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \qquad \lambda^{-1} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \mathcal{H}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{3-11}$$

A result of this is not only that every world point can be mapped to a pixel coordinate, but the opposite transformation can be done too. The homography matrix $\mathcal{H}$ does, however, make the assumption that the height of the point is zero. The assumption is made to make the transformation from pixel coordinates to real world coordinates possible. The perspective model assumes that all light travels in a straight line to the optical centre of the camera. Without the assumption that all points have zero height, there is an infinitely large set of points that can correspond to one pixel location. This follows from the fact that the pixel corresponds to a light ray passing through the pixel to the optical centre, with the corresponding real world point being located anywhere on that line. The homography matrix thus solves that problem by requiring the point to be located at the intersection of the line and the X-Y world plane. This makes transformation from $(u, v)$ to $(X, Y)$ coordinates using $\mathcal{H}^{-1}$ possible.

Functions to obtain the homography $\mathcal{H}$ exist in every standard computer vision toolbox and are again based on the method of [50]. At the start of every experiment, the BARC starts on the grid that can be seen in Figure 3-4a. The world coordinates of the center each circle on the template are known and the homography matrix can be calculated. By doing this, the pose of the camera can be estimated before every run and errors due to variations in camera pose between runs are eliminated. In real life applications the camera will be fixed in a more sturdy manner and this should not be necessary. A good illustration of the power of the inverse homography matrix $\mathcal{H}$ can be made by applying it to all pixels in an image. When applied and scaled up correctly the image is transformed to a projection as if it was taken from above the vehicle. An example of this can be seen in Subsection 3-1-3.

The extrinsic parameters for a non-moving vehicle can be extracted from the initial homography matrix $\mathcal{H}_0$. If the extrinsic parameters are known, the angles by which the camera was rotated in the previously selected rotation sequence can also be extracted. This is done by decomposing the homography matrix into a intrinsic matrix and a part of the extrinsic matrix. Due to the assumption that $Z = 0$, the matrix $\mathcal{H}_0$ can be written as the product of the intrinsic matrix $\mathcal{K}$ and a truncated version of the base extrinsic parameter matrix $\mathcal{E}_0'$:

$$R_0 = R_x(\phi_0)\ R_y(\theta_0)\ R_z(\psi_0) \tag{3-12}$$

$$\mathcal{H}_0 = \underbrace{\begin{bmatrix} f_x & \alpha & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathcal{K}} \underbrace{\begin{bmatrix} | & | & | \\ r_{1,0} & r_{2,0} & t_0 \\ | & | & | \end{bmatrix}}_{\mathcal{E}'_0} \tag{3-13}$$

Where $\mathcal{E}'_0$ represents the truncated extrinsic parameter matrix of the resting vehicle and $r_{i,0}$ the $i^{\text{th}}$ column of the rotation matrix $R_0$. By normalizing the first two columns of $\mathcal{E}'_0 k$ and calculating a third column using the cross product, the rotation matrix $R_0$ can be obtained. The angles of the base pose of the camera $(\psi_0, \theta_0, \phi_0)$ can then be calculated. The mathematics to do this are listed in Appendix B-2. The resulting angles vary slightly per experiment, approximately are:

$$\psi_0 = 90.0^\circ \qquad \theta_0 = 0.3^\circ \qquad \phi_0 = -116.0^\circ \tag{3-14}$$

During operation of the vehicle, the estimated roll angle $\phi_k$ (and in a future stage the pitch $\theta_k$, if it would be estimated) can then be added to the base pose of the camera to update rotation matrix $R_k$ and calculate an updated homography matrix $\mathcal{H}_k$:

$$R_k = R_x(\phi_0 + \phi_k)\ R_y(\theta_0 + \theta_k)\ R_z(\psi_0) \tag{3-15}$$

$$\mathcal{H}_k = \underbrace{\begin{bmatrix} f_x & \alpha & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathcal{K}} \underbrace{\begin{bmatrix} | & | & | \\ r_{1,k} & r_{2,k} & t_0 \\ | & | & | \end{bmatrix}}_{\mathcal{E}_k} \tag{3-16}$$

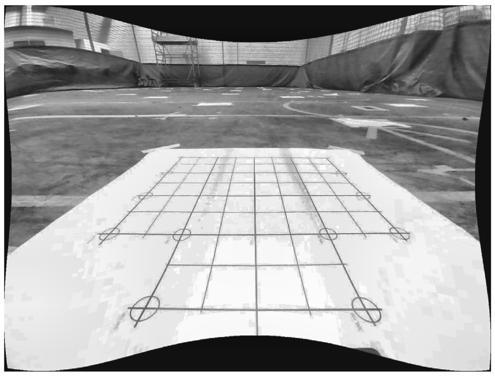
The updated homography matrix $\mathcal{H}_k$ is then used in the egomotion calculation of the CVA to perform the coordinate transformation on the tracked points from pixel location to real world coordinates.

## 3-2 Feature detection

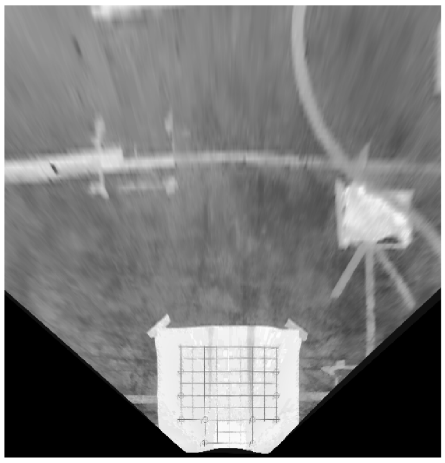
The next step in the CVA is to select points or patches in the image to follow. Often these areas of interest are referred to as features. Before features can be detected in the image, the image needs to be preprocessed to allow better feature detection.

### 3-2-1 Image preprocessing

When the images were captured during the experiments, all contrast, brightness and sharpness settings in the camera were set to a neutral position. The advantage of doing this is that the right amount of contrast can always be added later, since it is just a digital filter. If too



**(a)** Undistorted and contrast optimized image from BARC camera



**(b)** Same image projected on X-Y coordinate system using the found homography, but only bottom part showed for compactness

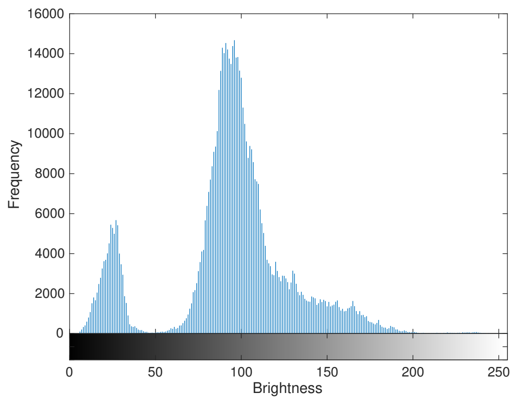
**Figure 3-4:** Comparison of an original undistorted frame (left) and part of the top down projection of the same frame, using the calibrated homography matrix $\mathcal{H}$

much contrast is added when capturing the image, the contrast is burned in and impossible to remove later.

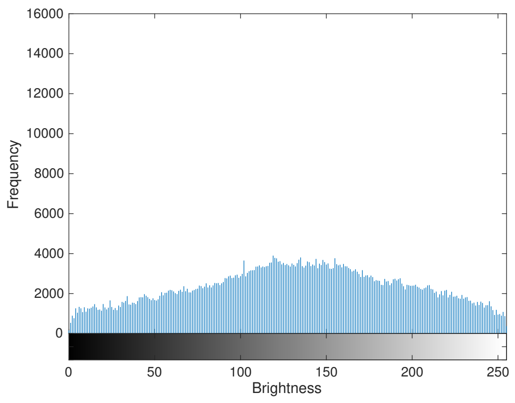
For reasons that are discussed later, the image tracking will be done on a black and white image. In Figure 3-6 an undistorted black and white image captured with the camera of the Berkeley Autonomous Race Car (BARC) during an experiment is shown. The tracking algorithm must identify distinguishable points to follow on the ground between two frames. Good points to follow usually are well distinguishable. To enable better tracking, the contrast of the image is enhanced using Contrast Limited Adaptive Histogram Equalization (CLAHE) [52]. Contrast can be expressed as the difference between very bright and very dark pixels in an image.

A black and white image can be interpreted as a two dimensional array with the value of each element representing the brightness of a pixel. For cheaper cameras, the brightness usually varies from 0 to 255, which represents 8 bits of brightness ($2^8 = 256$). To analyze the contrast of an image, the histogram of all pixel brightness values is a useful tool. In Figure 3-5a the histogram for the unprocessed image in Figure 3-6 can be seen. A lot of pixels have a brightness between 75 and 125. The goal of the preprocessing is to redistribute the pixel values in this very concentrated peak to a better spread set of pixel values, which is also called histogram equalization. Many different techniques for this exist. One often used to enhance contrast in surfaces is CLAHE.

CLAHE was originally used in fighter jet cockpits but is widely spread technique. CLAHE redistributes the histogram to cover the whole spectrum, but adds two interesting features. The first is that it divides the picture in a grid of tiles, usually 8 by 8, and calculates a histogram equalization function based on the center of each tile. It then redistributes all pixel values by bilinearly interpolating the equalization functions for all pixel. Bilinear interpolation



**(a)** Unprocessed image histogram



**(b)** Processed image historgram

**Figure 3-5:** Histogram for the unprocessed (left) and processed (right) image from Figure 3-6

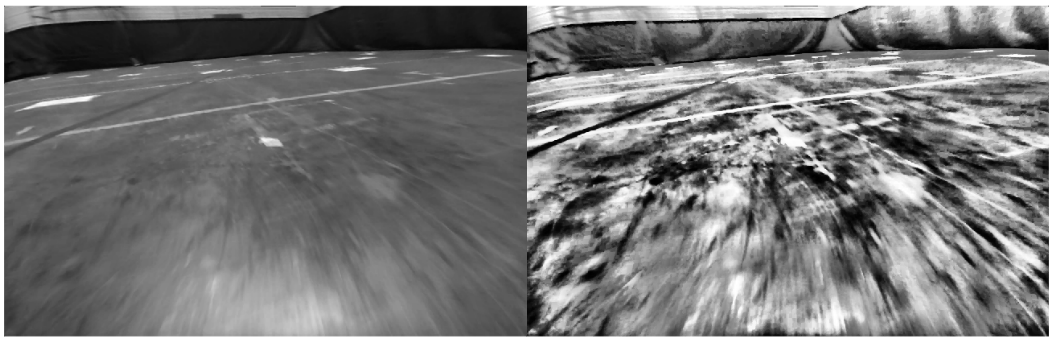


**Figure 3-6:** Comparison of a black and white image before (left) and after (right) CLAHE was applied. Used settings: `clip limit` $= 20.0$, `tile grid` $= [8, 8]$

acts the same as linear interpolation, but over two axes. Increasing the amount of tiles in an image sometimes allows for better contrast enhancement, but also increases computational load. Too much tiles will start to propagate noise, since having a lot of tiles will result in a small tile size. Each tile wants to map the pixels within the tile from dark to bright.

The second interesting feature is that when calculating the equalization function for each tile, it allows for a parameter that can determine by how much the contrast is enchaned. This is done by the `clip limit` parameter. Its exact workings are not relevant for this thesis, but it can be regarded as an amplification factor for the contrast in the image. The settings used for the amount of tiles in the image as well as the clip limit are discussed in Section 3-5. The result of applying CLAHE with the final optimal settings found in that section on a raw image can be seen in Figure 3-6.

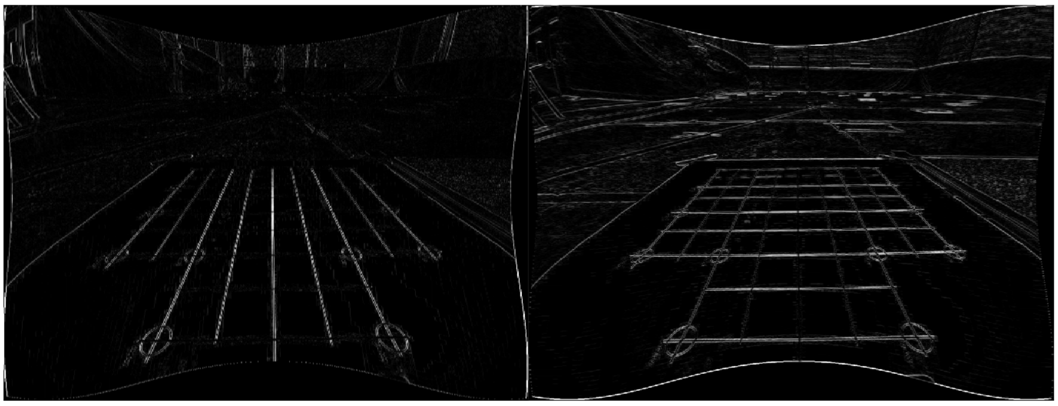


**Figure 3-7:** The horizontal gradient (left) and vertical gradient (right) computed for Figure 3-2b using the windows from (3-18)

### 3-2-2 Corner detection

Not every point inside an image is suited for tracking. Many different ways of detecting features exist and all of them have different applications where they are most efficient. For the tracking algorithm that will later be used, Lucas-Kanade optical flow, a corner detector based on the image gradient provides the best tracking accuracy [53]. This will probably be because both the feature detector and tracking algorithm then rely on the image gradient as core working mechanism. In an image, the image gradients can be visualized as the change of brightness for the image in the u- and v-direction. The image gradients in the u- and v-direction are defined as:

$$\nabla I(u,v) = \begin{bmatrix} \frac{\partial I}{\partial u}(u,v) & \frac{\partial I}{\partial v}(u,v) \end{bmatrix} \tag{3-17}$$

In images, which are discrete instances, it is usually calculated by convolution of the image with a window function, for example the ones from (3-18). An example of the u- and v-gradient for Figure 3-2b can be seen in Figure 3-7.

$$w_{du} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad w_{dv} = w'_{du} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{3-18}$$

For a corner to be distinguishable, it is desired that the change in brightness is large from the point itself to the points in its neighborhood. This can be visualized using a window function. This can be any window function, for example a square window or a Gaussian window. Its purpose is to select which pixels are considered within the neighborhood and can also weigh some pixels more than others. The window function will be called $w(u,v)$. For a point, the intensity (or brightness) of the point and its neighborhood can be quantified as:

$$I_w(u_p, v_p) = \sum_{u=u_{\min}}^{u_{\max}} \sum_{v=v_{\min}}^{v_{\max}} w(u,v) I(u,v) \tag{3-19}$$

Where $(u_p, v_p)$ is the point in the centre of the window, $(u_{\min}, u_{\max})$ and $(v_{\min}, v_{\max})$ are the borders of the window, $I(u,v)$ is the intensity at the point $(u,v)$ and $I_w(u_p, v_p)$ the weighted window intensity. Now the error of moving the window by a displacement $(x,y)$ can be considered.

$$E(x,y) = \sum_{u=u_{\min}}^{u_{\max}} \sum_{v=v_{\min}}^{v_{\max}} w(u,v)\left[I(u+x, v+y) - I(u,v)\right]^2 \tag{3-20}$$

Where $E(x,y)$ is the sum of squared errors for displacing the centre point of the window by $(x,y)$. For a good corner to follow, this error will be large. It was stated before that a good corner has a large change in brightness in both directions. To maximize this error, a Taylor series expansion was made:

$$I(u+x, v+y) \approx I(u,v) + I_x(u,v)x + I_y(u,v)y \tag{3-21}$$

$$E(x,y) \approx S(x,y) = \sum_{u=u_{\min}}^{u_{\max}} \sum_{v=v_{\min}}^{v_{\max}} w(u,v)\left[I_x(u,v)x + I_y(u,v)y\right]^2 \tag{3-22}$$

Where $I_x$ and $I_y$ are partial partial derivatives of $I$ in the subscript direction and $S(x,y)$ is the Taylor series approximation of the window displacement error $E(x,y)$. (3-22) can be written in a matrix form:

$$S(x,y) \approx \begin{bmatrix} x & y \end{bmatrix} \left( \sum_{u=u_{\min}}^{u_{\max}} \sum_{v=v_{\min}}^{v_{\max}} w(u,v) \underbrace{\begin{bmatrix} I_x(u,v)^2 & I_x(u,v)I_y(u,v) \\ I_x(u,v)I_y(u,v) & I_y(u,v)^2 \end{bmatrix}}_{C} \right) \begin{bmatrix} x \\ y \end{bmatrix} \tag{3-23}$$

A strong corner should consequently maximize the matrix $C$, which is sometimes called the corner matrix. The matrix $C$ gives an indication of the change in brightness if a unit step is taken in the direction of $(x,y)$. The eigenvalues of $C$ indicate the direction of steepest ascent and the steepest descent/smallest ascent of the surface that represents the change in brightness. This leads to the conclusion that if both eigenvalues are large, the brightness of the image changes by a large amount in all directions at that point of the image. Two large eigenvalues will therefore indicate that the point can be considered a corner. A demonstration of this can be seen in Figure 3-8.

The above described selection mechanism, looking at the minimum eigenvalue of the $C$ matrix, was introduced in the work of Shi-Tomasi [54]. The quality of a point $Q_p$ is considered to be equal to the minimum eigenvalue of the $C$ matrix:

$$Q_p = \min(\lambda_1, \lambda_2) \tag{3-24}$$

Where $\lambda_i$ represents the i$^{\text{th}}$ eigenvalue of the $C$ matrix. By posing a lower bound on $Q_p$ or selecting the best $N$ points out of all points in the image, a group of corners can be found in the image. In this project the 500 points with the highest minimum eigenvalue were detected at each frame by the CVA. This number was chosen as a upper bound for the algorithm, but was almost never reached during driving.

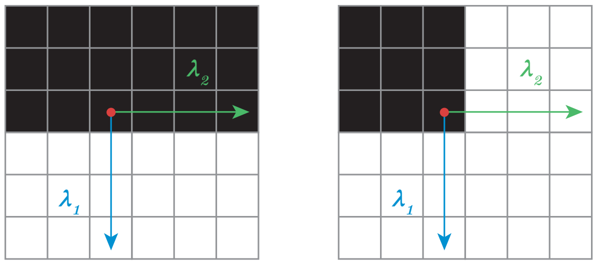


**Figure 3-8:** Visual representation of an edge and a corner. An edge has a large change in brightness over one direction ($\lambda_1$ large, $\lambda_2$ small) and a corner has a large change in brightness over two directions (both $\lambda_1, \lambda_2$ large).

After that outlier detection was performed by rejecting all points that had its corner quality not within 10% of the quality of the strongest corner of that particular frame. Then the points were passed on to the tracking algorithm. At every new frame all previous points were discarded and a new set of points was detected again.

**Region of Interest** Not every part of the image is fit for tracking. As can be seen in Figure 3-9 only a part of the image is selected to be used in the corner detection and optical flow. The black zones created by the undistortion step are used as a starting point (red[1]). Then a margin of 40 pixels inwards is used (blue) to make sure that all functions run smoothly, since the corner detection algorithm does not like running close to the border of an image. The blue zone also blocks the top 100 pixels since they are above the horizon and tracking on the ground plane is not applicable. Finally, a small set of pixel of 40 pixel high in the $v$-direction (green) is blocked from tracking since it describes a large distance in a very small set of pixels. In world coordinates 11 meters are described in the $Y$-direction by just 40 pixels in the $v$-direction. To put into perspective: the remaining 500 pixels in the $v$-direction describe 1.5 meters in the $Y$-direction in world coordinates.

## 3-3 Optical flow

The algorithm is now able to detect corners that are suitable for tracking between two frames. For following points, sparse and dense optical flow are two powerful yet not too complex tools. Other advanced techniques such as ORB-SLAM also exist. However, the goal during this thesis was always that the algorithm should be able to run online on the test platform in the near future. The optical flow algorithms and especially sparse optical flow are much less computationally intensive. This is a great advantage since a frequency as high as possible is desired. For the BARC this will be 30 Hz, because of the sampling rate of the camera. The object the camera of the BARC is following, a ground plane, has a uniform velocity. However, due to the pose of the camera, the velocity is not perceived as uniform across the ground plane. The pixels closer to the camera appear to be moving faster then the pixels far away from the camera. Cases where there are no large areas having the same velocity, sparse optical flow is preferred over dense optical flow. By employing sparse optical flow, only several

[1]The colors red, blue green are chosen because of their clarity and distinguishable property when print. This has nothing to do with the RGB color channels of an image.

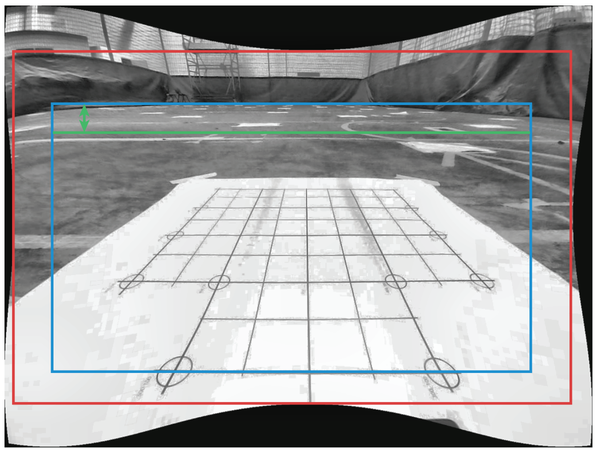


**Figure 3-9:** The three steps of selecting the Region of Interest for the detection and tracking algorithm. Red marks the outer bound of the undistorted image, blue marks the 40 pixels border for the tracking algorithm and green marks the extra blocking of the small bar of pixels that describes a very large set of longitudinal distances.

points are tracked between frames. A nice side effect of this is that it is also computationally less intensive.

## 3-3-1 Lucas Kanade optical flow

Sparse optical flow is most often calculated using the method proposed by Lucas and Kanade [55]. In this section the operating principles of the original algorithm and its extension to an iterative pyramidal implementation will be discussed. The original Lucas Kanade optical flow algorithm starts by making three assumptions:

1. The change of intensity of a point in the image between two frames is low.
2. Neighboring pixels all have the same motion.
3. The movement of a pixel is not too big.

Consider a pixel at location $(u, v)$ at time $t$ with intensity $I(u, v, t)$. The image gradients of the image are known, namely $I_u(u, v)$ and $I_v(u, v)$. By assuming that the intensity of a *point* displayed in an image remains the same between time steps, a change in intensity of one particular *pixel* at a fixed location $(u, v)$ must be the result of the scene moving. Because the image gradients are known, the temporal change in intensity at a fixed *pixel* location must be the result of a displacement by the vector $(\Delta u, \Delta v)$. The change in intensity due to displacement can be approximated by multiplying the image gradient in each direction with the displacement in that respective direction. The temporal change in intensity can be approximated by multiplying the time derivative of the image intensity with the time step taken. Approximation is based on the first order Taylor series expansion. This yields the following equation:

$$I_u(u, v, t)\Delta u + I_v(u, v, t)\Delta v = -I_t(u, v, t)\Delta t \tag{3-25}$$

This equation is called the optical flow equation, where $I_t(u, v, t)$ is the partial derivative of the intensity at $(u, v)$ with respect to time. For this project, tracking is only done between two subsequent frames, which makes $\Delta t = 1$ and is therefore left out. Unfortunately it is impossible to solve for $[\Delta u, \Delta v]$, the desired displacement, for one point alone. This can be worked around by using the second and third assumption. By applying (3-25) to a patch of points around the point $(u, v)$, again a so called window, a set of equations is obtained that can be solved as a least squares problem. A window of size $(2\omega_u + 1, 2\omega_v + 1)$ with $(u, v)$ in the centre of it can track motion up to $\omega_u$ in the $u$-direction and $\omega_v$ in the $v$-direction. The full set of equations and the solution for $(\Delta u, \Delta v)$ is of the following form:

$$\underbrace{\begin{bmatrix} I_u(u-\omega_u, v-\omega_v) & I_v(u-\omega_u, v-\omega_v) \\ I_u(u-\omega_u, v-\omega_v+1) & I_v(u-\omega_u, v-\omega_v+1) \\ \vdots & \vdots \\ I_u(u+\omega_u, v+\omega_v) & I_v(u+\omega_u, v+\omega_v) \end{bmatrix}}_{A\in\mathbb{R}^{(2\omega_u+1)(2\omega_v+1)\times 2}} \underbrace{\begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix}}_{x\in\mathbb{R}^{2\times 1}} = -\underbrace{\begin{bmatrix} I_t(u-\omega_u, v-\omega_v) \\ I_t(u-\omega_u, v-\omega_v+1) \\ \vdots \\ I_t(u+\omega_u, v+\omega_v) \end{bmatrix}}_{b\in\mathbb{R}^{(2\omega_u+1)(2\omega_v+1)\times 1}} \tag{3-26}$$

$$x = (A^T A)^{-1} A^T b \tag{3-27}$$

$$\begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \begin{bmatrix} \sum I_u^2 & \sum I_u I_v \\ \sum I_u I_v & \sum I_v^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum I_u I_t \\ -\sum I_v I_t \end{bmatrix} \tag{3-28}$$

In this way the displacement of one point at one time step can be calculated. This procedure needs to be repeated for all detected points $[P_{1,k} \dots P_{N_p,k}]$ at every time step $k$. As already stated, the window size gives a limitation on the biggest movement $(\Delta u, \Delta v)$ possible, which reflects in the third constraint. Normal values for the window parameter range from 2 to 7. However, when looking at the average speeds the vehicle reached during experiements, a longitudinal speed of $3\,\mathrm{m\,s^{-1}}$ and a lateral speed of $2\,\mathrm{m\,s^{-1}}$, this gives problems. As said before, the pixels closest to the camera will give the greatest displacement when the ground plane is moving. For a longitudinal speed of $3\,\mathrm{m\,s^{-1}}$ this results in a displacement of 200 pixels between two subsequent frames along the $u$-axis and for the lateral speed of $2\,\mathrm{m\,s^{-1}}$ it results in a displacement of 150 pixels between two subsequent frames along the $v$-axis. To accommodate for these movements, a window of $(200 \times 150)$ pixels would be needed. This is undesirable for numerous reasons, mostly computer power and violation of assumption two and three.

## 3-3-2 Iterative pyramidal implementation

The Iterative Pyramidal Lucas Kanade Optical Flow (IPLKOF) algorithm can solve this problem by using downsampled versions of the image stacked into so called image pyramids, together with an iterative approach [56]. The image pyramid starts off at level $L = 0$, the original image, and downsamples each subsequent level by a factor 2 on both axis. The lower resolution of the downsampled images combined with the same window size allows for tracking of large motions on a big scale first, then refining the estimate by going down the

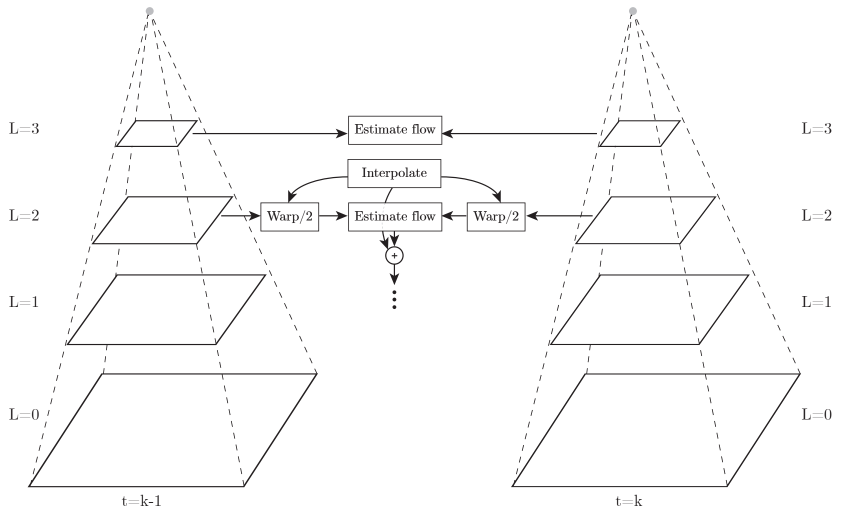


**Figure 3-10:** Visual representation of the pyramidal Lucas Kanade optical flow level. The algorithm here has a maximum pyramid level of $L_m = 3$. Each level is downsampled by a factor 2 in both directions. The algorithm starts at the highest level, estimates the flow and then goes a level down. By warping and interpolating the results from previous levels are maintained. Image adapted from [57]

**Table 3-2:** Possible combinations of highest pyramid level and minimum required window size to track a movement of $3\,\mathrm{m\,s^{-1}}$ longitudinally

| Max level | $\omega_{\min}$ |
|---|---|
| $L_m = 0$ | 200 pixels |
| $L_m = 1$ | 67 pixels |
| $L_m = 2$ | 29 pixels |
| $L_m = 3$ | 14 pixels |
| $L_m = 4$ | 7 pixels |
| $L_m = 5$ | 4 pixels |

pyramid to lower levels (thus larger images) and allowing for finer tracking. The iterative approach reflects in the fact that the larger motion of the high, low resolution, levels is used as a starting point for the finer motions on the lower levels. This is illustrated in Figure 3-10. For a full implementation of the algorithm see [56]. The maximum displacement that can be tracked by a symmetric window with size parameter $\omega$ equals:

$$d_{\max} = (2^{L_{\max}+1} - 1)\,\omega \tag{3-29}$$

To obtain a maximal displacement of 200 pixels, several combinations of maximum level $L$ are possible. These are listed in Table 3-2. Later, in Section 3-5 the best combination of parameters will be evaluated.

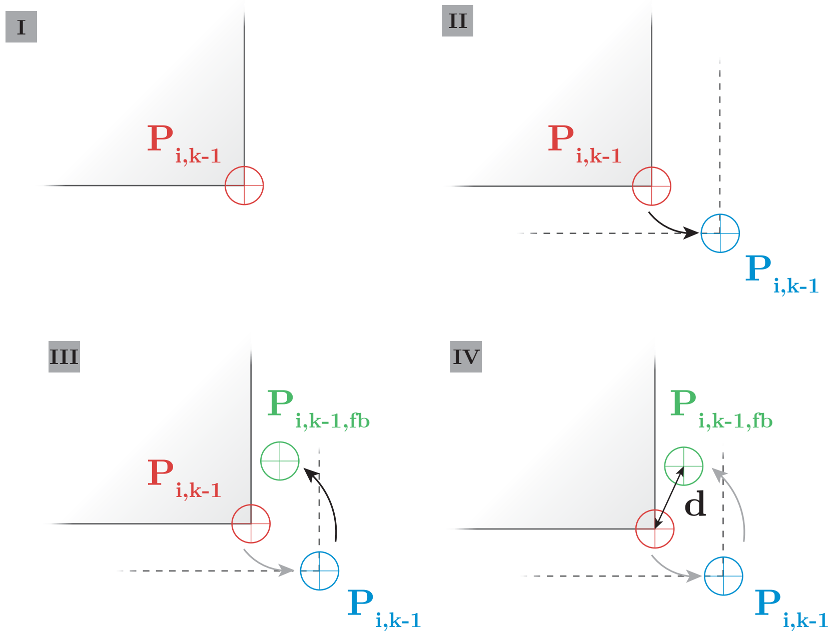


**Figure 3-11:** The four stages of the tracking algorithm: I. Corner detection using Shi-Tomasi's algorithm II. Forward tracking using Pyramidal Lucas Kanade (Pyr-LK) optical flow III. Backwards tracking using IPLKOF IV. Inspection of forward-backward error, track is valid is $d < d_{max}$

## 3-3-3 Outlier rejection

To complete the optical flow algorithm, some outlier rejection is performed. In [58], the authors propose running the Lucas Kanade optical flow algorithm backwards after normal tracking to obtain a measure of tracking accuracy. The forward-backward error outlier rejection works in four steps and is also shown in Figure 3-11:

1. Start with a set of points $[p_{1,k-1} \dots p_{N_p,k-1}]$ detected in frame $k-1$ using Shi-Tomasi corner detection method, reject corners with quality below the 10% threshold.

2. Estimate position of all points $[\tilde{p}_{1,k} \dots \tilde{p}_{N_p,k}]$ in frame $k$ by applying IPLKOF from frame $k-1$ to frame $k$ to $[p_{1,k-1} \dots p_{N_p,k-1}]$. A tilde indicates a coordinate produced by IPLKOF.

3. Estimate forward-backward position of all points $[\tilde{p}_{1,k-1} \dots \tilde{p}_{N_p,k-1}]$ by applying IPLKOF from frame $k$ to $k-1$ to $[\tilde{p}_{1,k} \dots \tilde{p}_{N_p,k}]$.

4. Calculate for every point if the forward-backward projection error $|p_{i,k-1} - \tilde{p}_{i,k-1}|$. Point pare $(p_{i,k-1}, \tilde{p}_{i,k})$ is an inlier if $|p_{i,k-1} - \tilde{p}_{i,k-1}| < d_{\max}$.

A setting of $d_{\max} = 0.5$ gave adequate results. For the points with less than 0.5 pixel of FB-error, the amount of FB-error is later used as one of the inputs for the Dynamic Covariance Estimation (DCE).

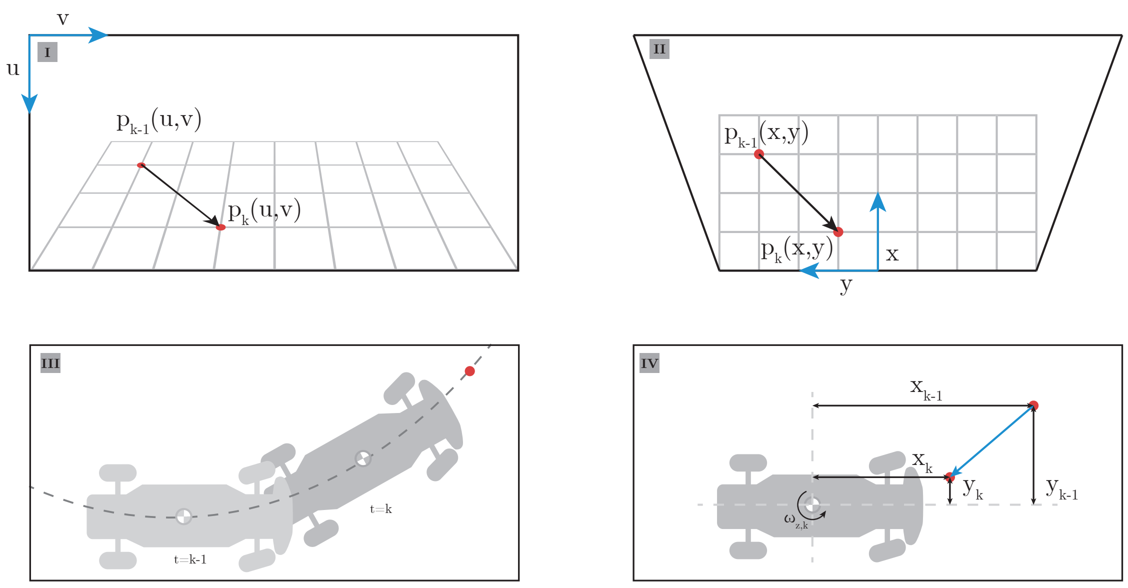


**Figure 3-12:** Visual representation of the essential steps of the tracking algorithm: I. Point tracking along the ground plane in an image frame II. Transformation from image coordinates $(u, v)$ to real world coordinates $(x, y)$ III. Assumption that a moving point in the image actually corresponds to a moving vehicle IV. Calculation of vehicle speed by displacement of point

## 3-4 Egomotion calculation

For each frame $k$, the above described corner detection and optical flow algorithm will yield a total of $N_p$ points that are described with its pixel location in previous frame $k-1$ and current frame $k$. Using the homography that was described in Section 3-1 the real world positions of both sets of points can be estimated. The displacement of the origin of the world coordinate system, which conveniently enough is located below the centre of gravity of the vehicle, can be calculated by applying kinematic formulas. For each tracked point a separate velocity will be calculated. Consider the collection of points $[p_{1,k} \dots p_{N_p,k}]$, the speed of the vehicle according to each point $p_{i,k}$ in the longitudinal and lateral direction will be:

$$V_{x,i}(k) = -\left(\frac{p_{x,i}(k) - p_{x,i}(k-1)}{\Delta t} + \omega_z(k)\, p_{y,i}(k-1)\right) \tag{3-30}$$

$$V_{y,i}(k) = -\left(\frac{p_{y,i}(k) - p_{y,i}(k-1)}{\Delta t} - \omega_z(k)\, p_{x,i}(k-1)\right) \tag{3-31}$$

This will yield the vector of velocities that was described in Section 2-3. A final overview of the whole tracking and speed calculation process is given in Figure 3-12. In algorithm 1 an overview of the algorithm in pseudo code is also given.

```
Result: Vehicle egomotion $[V_{x,i,k}, V_{y,i,k}]$ for every point $[p_{1,k}, p_{2,k}, \dots, p_{i,k}, \dots p_{N_{p,k},k}]$ in
        frames $[F_1, F_2, \dots, F_k, \dots, F_{k_{max}}]$
Initialization;
Capture, undistort and equalize frame $F_1$;
for $k \leftarrow 2$ to $k_{max}$ do
    Detect points $[p_{1,k-1} \dots p_{N_p,k-1}]$ in $F_{k-1}$ for which
     $Q_{p_i} > \Big(0.1 \max([Q_{p_{1,k-1}} \dots Q_{p_{N_p,k-1}}])\Big)$;
    Capture, undistort and equalize $F_k$ ;
    $[\tilde{p}_{1,k}, \dots, \tilde{p}_{N_p,k}] \leftarrow$ track detected points $[p_{1,k-1}, \dots, p_{N_p,k-1}]$ from $F_{k-1}$ to $F_k$;
    $[\tilde{p}_{1,k-1}, \dots, \tilde{p}_{N_p,k-1}] \leftarrow$ track predicted points $[\tilde{p}_{1,k}, \dots, \tilde{p}_{N_p,k}]$ from $F_k$ to $F_{k-1}$ ;
    forall $\tilde{p}_{i,k}$ with $|p_{i,k-1} - \tilde{p}_{i,k-1}| < d_{max}$ do
        $P_{i,k-1}(x_{i,k-1}, y_{i,k-1}) \leftarrow [h_x^{-1}(p_{i,k-1}), h_y^{-1}(p_{i,k-1})]$ ;
        $\tilde{P}_{i,k}(x_{i,k}, y_{i,k}) \leftarrow [h_x^{-1}(\tilde{p}_{i,k}), h_y^{-1}(\tilde{p}_{i,k})]$ ;
        $V_{x,i,k} \leftarrow (\tilde{P}_{i,k,x} - P_{i,k-1,x})/\Delta t + \omega_{z,k} P_{i,k-1,y}$ ;
        $V_{y,i,k} \leftarrow (\tilde{P}_{i,k,y} - P_{i,k-1,y})/\Delta t - \omega_{z,k} P_{i,k-1,x}$ ;
    end
end
```

**Algorithm 1:** Computer Vision Algorithm overview

## 3-5 Parameter tuning

The final step of creating the Computer Vision Algorithm (CVA) is setting the parameters of the different functions to the optimal value. There are many parameters that can be set. Some have a greater effect on the performance of the algorithm than others. To select which combination of parameters performs best, a sensitivity analysis was performed for the parameters that were considered to have the greatest influence on performance. To evaluate the optimal performance, the median speed per frame of the calculated velocities from the CVA was compared to a validation speed obtained by an accurate external Motion Capture System (MCS). The MCS is described in Section 5-3. To assess the performance, the Root Mean Square Error (RMSE) and Variance Accounted For (VAF) were used. The two are defined as:

$$\text{VAF} = \left(1 - \frac{\text{Var}(x - \hat{x})}{\text{Var}(x)}\right) \cdot 100\% \tag{3-32}$$

$$\text{RMSE} = \sqrt[2]{\frac{\sum_{i=1}^{n}(x_i - \hat{x}_i)^2}{n}} \tag{3-33}$$

The VAF gives an indication of how well the estimated signal follows the changes in validation signal. The RMSE gives an indication deviation from the validation signal. The VAF was limited to a lower bound of 300% in the function calculating it. Values of 300% in any table can actually be lower then 300%.

### 3-5-1 CLAHE parameters

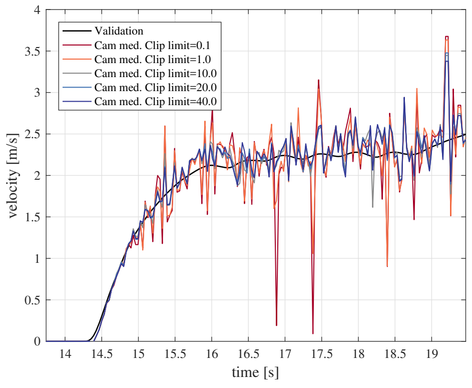
First the most important parameters for the CLAHE method were evaluated. The results of varying the clip limit can be seen in Figure 3-13. It can be seen that increasing the clip limit boosts performance when going from 0.1 to 10 and the median camera velocity gets closer to the validation velocity. Especially the spikes start to disappear. After a clip limit of 10 the improvement flattens out. This is also reflected in the performance metrics listed in Table 3-3. The RMSE gets to $0.20\,\mathrm{m\,s^{-1}}$ and $0.10\,\mathrm{m\,s^{-1}}$ for respectively $V_x$ and $V_y$ and does not improve any further by raising the clip limit. During this analysis the tile grid size was kept at $[16, 16]$ tiles. The clip limit was chosen to be set at 20 for the CVA.

Then the tile grid size was evaluated. It can be seen in Figure 3-14 that increasing the tile grid size gives an initial improvement. In the plots it seems to be the case that improving the tile grid size only has positive effects. The median camera velocity only gets close to the validation speed and spikes disappear. When looking at the performance metrics in Table 3-3 it can be seen that a grid of $[16, 16]$ tiles gives slightly better results at the VAF than $[8, 8]$ tiles. The clip limit was kept constant at 20 for these experiments. The final tile grid size was set at $[16, 16]$.

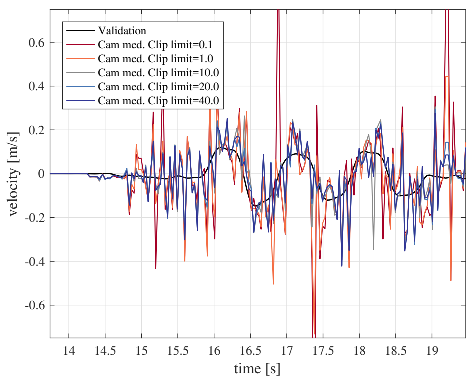
A visual explanation for the found settings can be seen in Figure 3-16. There a mosaic can be seen of the combinations of clip limit and tile size used in the sensitivity analysis and their effect on the output image. It can be seen that as the tile grid size increases initially the contrast increases. After the third column ([16, 16] tiles), the image starts to fall apart. The tiles become so small that the noise and small variations in the surface start getting amplified too much. A higher clip limit seems to provide a better contrast (and therefore better tracking) without any noticeable downsides.

### 3-5-2 Optical Flow parameters

After the two CLAHE parameters were tuned, the window size and maximum pyramid level of the IPLKOF was evaluated. The CVA was run with all viable combinations from Table 3-2. It can be seen that if the window level is low, the algorithm has difficulties in tracking longitudinal velocity, where the result is just too low, and has some strange variations in lateral velocity. Once the window level starts to increase the results get pretty close to the validation speed. The best level settings were those with maximum level 3 and 4. This can be explained by the fact that a higher level allows for a smaller window, thus increasing accuracy. Apart from computation time there are not many downsides to increasing the maximum level. After level 3 and 4 the downsampled image becomes really small, $1/8^{\mathrm{th}}$ and $1/16^{\mathrm{th}}$ of the original resolution per axis, so higher levels don't lead to better results. For the CVA a maximum level of 3 and a window of $[14, 14]$ was used. It was also tested if keeping the maximum level at 3 and varying the window size improved the results, but this was not the case. These test are therefore also not shown.

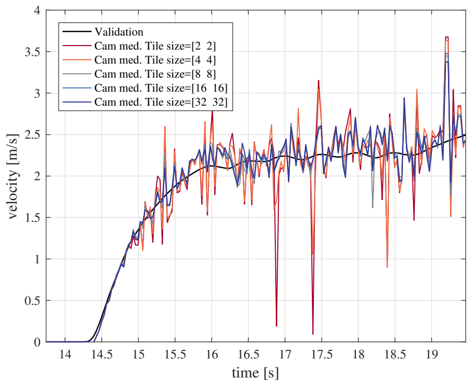


**(a)** Median longitudinal camera speed

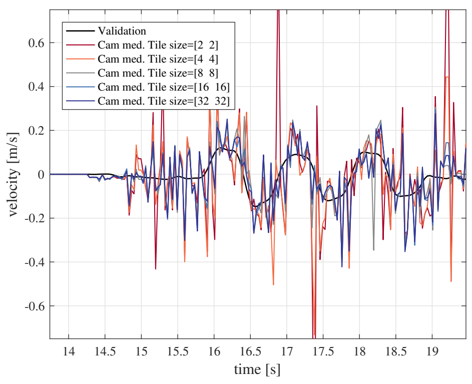


**(b)** Median lateral camera speed

**Figure 3-13:** Sensitivity plot for the clip limit setting of CLAHE. 5 variations of the clip limit were used to run the CVA and results are plotted against the validation speed measured by the MCS

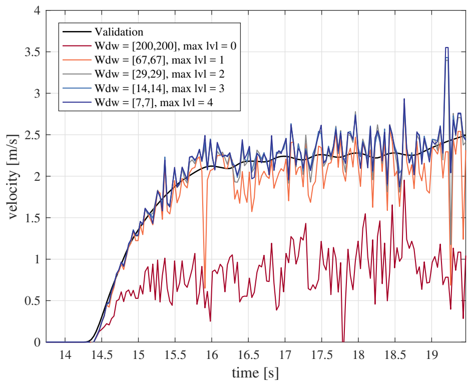


**(a)** Median longitudinal camera speed

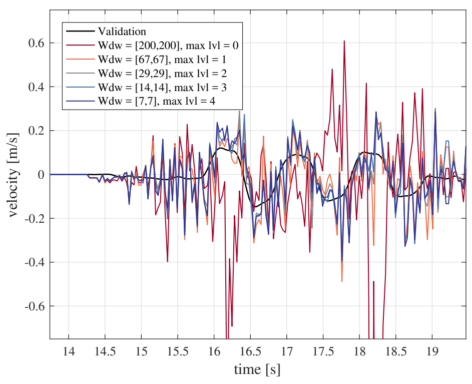


**(b)** Median lateral camera speed

**Figure 3-14:** Sensitivity plot for the tile grid size of CLAHE. 5 variations of the grid size were used to run the CVA and results are plotted against the validation speed measured by the MCS



**(a)** Median longitudinal camera speed



**(b)** Median lateral camera speed

**Figure 3-15:** Sensitivity plot for the window size and maximum pyramid level of the Iterative Pyramidal Lucas Kanade Optical Flow. 5 combinations of maximum level and window size were used to run the CVA and results are plotted against the validation speed measured by the MCS

**Table 3-3:** Performance metrics for tile grid size and clip limit parameter sensitivity analysis. All analysis performed on maneuver 3C.

| | Camera median $V_x$ | | Camera median $V_y$ | |
|---|---|---|---|---|
| | RMSE | VAF | RMSE | VAF |
| Clip limit | | | | |
| 0.1 | 0.4103 | 83.34% | 0.2348 | -217.7% |
| 1 | 0.3294 | 86.87% | 0.1598 | -181.2% |
| 10 | 0.2238 | 87.62% | 0.1108 | -129.5% |
| 20 | 0.2149 | 88.58% | 0.1032 | -129.8% |
| 40 | 0.2086 | 78.10% | 0.1019 | -234.5% |
| Tile grid size | | | | |
| [2 2] | 0.2441 | 51.16% | 0.1215 | -300.0% |
| [4 4] | 0.2198 | 68.79% | 0.1142 | -300.0% |
| [8 8] | 0.2149 | 86.49% | 0.1032 | -164.3% |
| [16 16] | 0.2149 | 88.58% | 0.1032 | -129.8% |
| [32 32] | 0.2745 | 88.33% | 0.1246 | -123.7% |

**Table 3-4:** Performance metrics for IPLKOF window and max level parameter sitivity analysis. All analysis performed on maneuver 3C.

| Max pyramid level | Window size | Camera median $V_x$ | | Camera median $V_y$ | |
|---|---|---|---|---|---|
| | | RMSE | VAF | RMSE | VAF |
| 0 | [200 200] | 1.2502 | 36.86% | 0.4987 | -300.0% |
| 1 | [67 67] | 0.3123 | 75.90% | 0.1242 | -229.8% |
| 2 | [29 29] | 0.2322 | 85.03% | 0.1044 | -133.6% |
| 3 | [14 14] | 0.2149 | 88.58% | 0.1032 | -129.8% |
| 4 | [7 7] | 0.2215 | 87.23% | 0.0999 | -114.4% |

## 3-6 Conclusion

In this chapter the CVA was discussed. The CVA is used for generating vehicle egomotion measurements based on the images it sees of the ground plane. The algorithm detects corners in an image and tracks these corners between two frames. If the corners have a sufficiently small forward-backward projection error, they are used in the observer. Each set of original and tracked point is converted into real world coordinates using a homography matrix. This matrix is based on the pinhole camera model and is decomposed in three rotation matrices, so later a correction for roll and pitch can be done. Once the position of the original and tracked point is known in the real world, the longitudinal and lateral speed of the vehicle according to the displacement of the point is calculated. This is done for all corners at every time step. This yields a vector of 100 to 400 points, depending on how many good corners are found. Finally the sensitivity for some parameters in the CVA was discussed and an combination was selected for experiments.

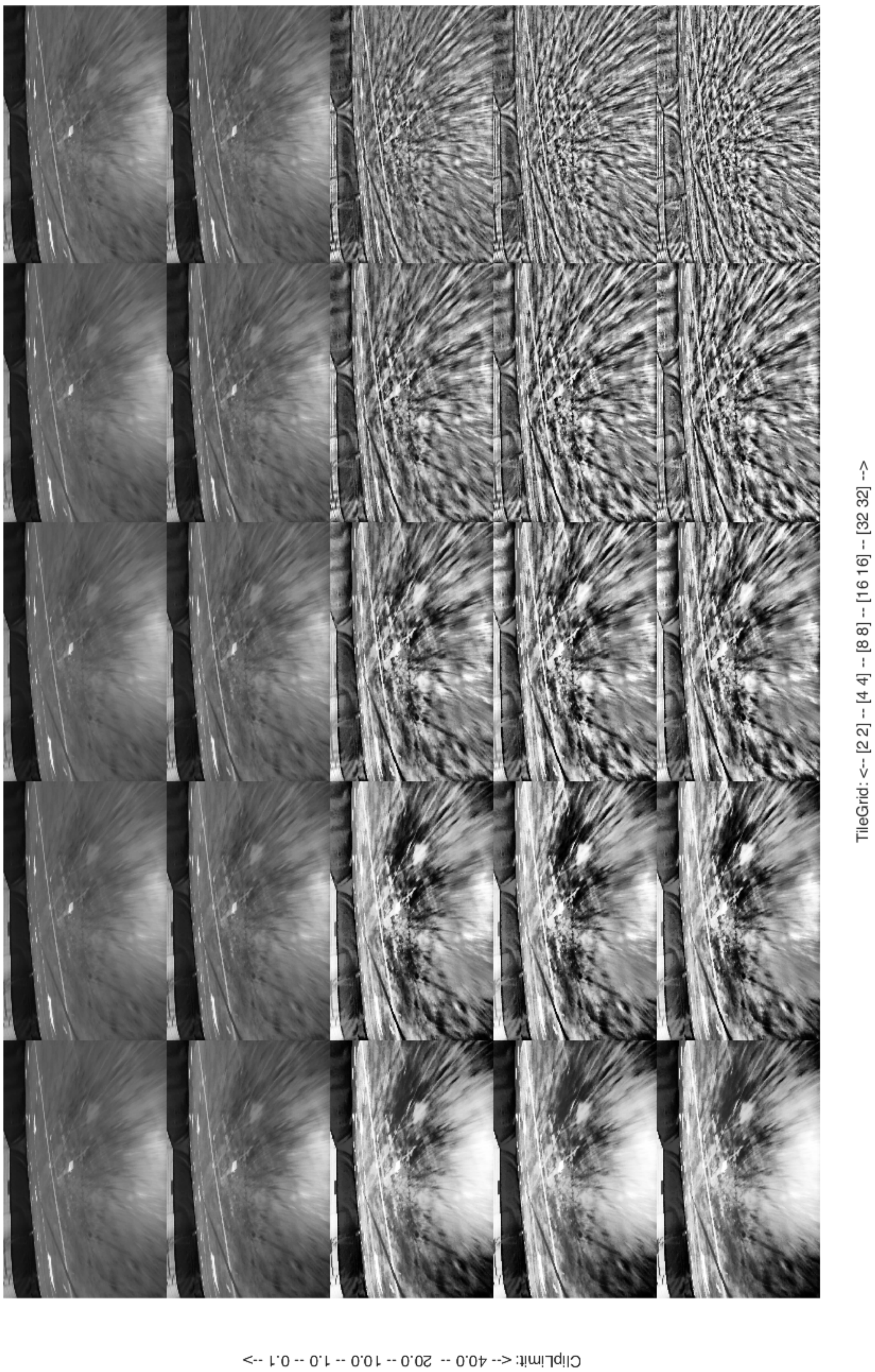

TileGrid: <-- [2 2] -- [4 4] -- [8 8] -- [16 16] -- [32 32] -->
ClipLimit: <-- 40.0 -- 20.0 -- 10.0 -- 1.0 -- 0.1 -->

**Figure 3-16:** Mosaic of 5 versions of the clip limit, 0.1-1.0-10.0-20.0-40.0 and five different tile grid sizes, 2×2-4×4-8×8-16×16-32×32.

# Chapter 4

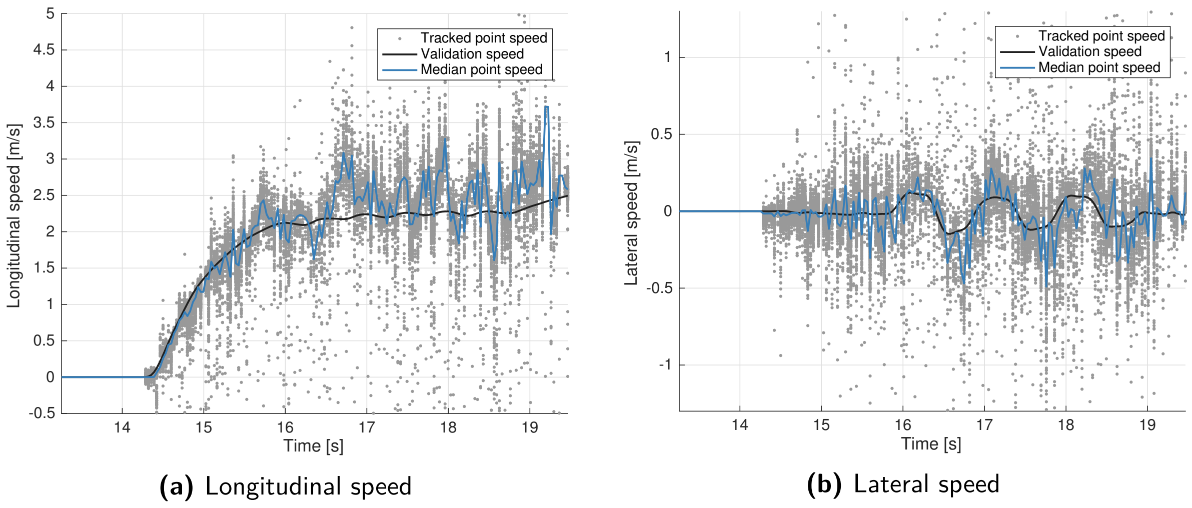
# Dynamic Covariance Estimation

In this chapter the methodology chosen for filtering the camera measurements will be discussed. First the requirements of the filter will be discussed. Then a further understanding of the sources of error will be created. Based on that understanding, heuristic functions are created that can adjust the measurement covariance matrix that is used by the Extended Kalman Filter (EKF). The complete set of functions and algorithms that adjust the covariance matrix is called the Dynamic Covariance Estimation (DCE).

## 4-1 Requirements

The Computer Vision Algorithm (CVA) described in the previous chapter can be used to estimate the velocity of a vehicle. Where in previous works red markers were placed on the ground plane to aid tracking, this was not done for this project. Although this resembles a more realistic environment, it decreases tracking accuracy. In Figure 4-1 the longitudinal velocity and lateral velocity for a maneuver consisting of constant rear wheel torque and a sinusoidal steering input with parameters $f_\delta = 1\,\text{Hz}, \delta_f = [-13°, 13°], F_{xR} = 2.2N$ can be seen. The maneuver is further described in Section 6-1, referenced as maneuver 3C. When comparing the speed of all tracked points per frame with the validation velocity it can be seen that there exists a large variance between calculated velocities from the tracked points at every time step. The validation velocity is obtained using an accurate external Motion Capture System (MCS), further described in Section 5-3.

Furthermore, it can be observed that a median filter filters the measurements properly in most of the frames, but at some frames it is completely off. Several spikes can be seen in the lateral speed signal, where all tracked points deviate by a large amount and so also the median deviates. Next to that, median filtering is a nonlinear operation, which is not preferable. It is consequently not only a problem of rejecting certain outliers per frame to get better results, but also one of weighing the whole camera measurement at every time step. As a result of this another approach to filtering the camera measurements is chosen in this project. To achieve



**(a)** Longitudinal speed

**(b)** Lateral speed

**Figure 4-1:** The unfiltered median longitudinal (left) and lateral (right) point speed compared with the validation speed. The maneuver performed consists of a sinusoidal steering input of 1 Hz with an amplitude of $20^\circ$ with a constant rear wheel torque (maneuver 3C).

both of the previously described goals, outlier rejection and overall camera signal weighing, all points are processed as a measurement signal in the EKF, as was described in 2-3.

The EKF uses an estimate of the process noise covariance $Q$ and measurement noise covariance $R$ to calculate an optimal observer gain. This requires both matrices to be known beforehand. In this chapter it is shown that the error of a point tracked in an image depends among other things on its position in the frame, the corner quality and forward backward tracking error. Therefore, it is impossible to set an adequate $R$ matrix *a priori* and keep it fixed during operation of the EKF. Online adaptation of the $R$ matrix is therefore required.

Online adaptation of the ($Q$ and) $R$ matrix is an idea that has been around for quite some time. It is most applicable when it is hard to determine the $Q$ and $R$ matrix and a solution is desired where the values of the matrices (asymptotically) converge to a value for optimal estimation performance [59, 60]. The problem of applying that concept to this project, is that at every time step a different amount of new points will be detected and tracked. No results from the previous frame are carried over. This results in a lot of new points being used at every frame with a new, different, location. In this chapter it is shown that the error on a calculated velocity, and thus its (co)variance, depends on the location of a pixel in the frame.

These conditions are analogous to selecting a random number of sensors with a random variance and covariance between them at every time step. Letting the $R$ matrix converge to the right value when the variance of every sensor and the number of sensors changes at every time step will be practically impossible. Therefore another approach of adaptation of the measurement noise covariance submatrix is used. In that approach the variance of the error on one measurement is modelled as a stochastic process influenced by various heuristic functions. These functions are fitted *a priori* based on correlation between certain parameters and the squared velocity error of points. Using this model of one variance the full covariance submatrix can be scaled.

## 4-2 Error model

Before it is possible to create a function that adjusts the covariance submatrix of the camera measurements, the stochastic model for the measurement noise variance is needed first. The image tracking process is described in the previous chapter. For the covariance estimation algorithm the assumption is made that at three places an error is introduced. Two times an error is added when transforming from pixel to world coordinates and one time an error is added due to the tracking.

### 4-2-1 Homography error

The first type of error is the homography error. It is introduced when a image point is converted to a world point. For a point tracked from frame $k-1$ to $k$, the first moment this error surfaces is at sample $k-1$. The point $p_{k-1}(u_{k-1}, v_{k-1})$ at sample $k-1$ is transformed to a real world point $P_{k-1}(\tilde{x}_{k-1}, \tilde{y}_{k-1})$. This transformed point may contain a transformation error. Therefore the coordinates $(\tilde{x}_{k-1}, \tilde{y}_{k-1})$ are denoted with a tilde, representing the fact that they contain an error. This error can be caused by several different factors:

1. The pose of the camera may not exactly be equal to the one when the homography was calibrated, causing the real world point to shift. This can happen for example due to pitch, roll and heave of the vehicle. Although a framework to correct for this was discussed, the correction is not perfect and only done for the roll angle.

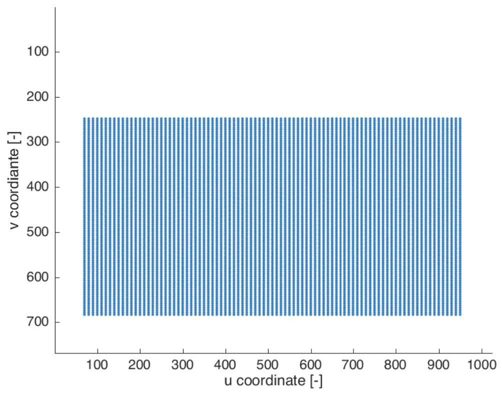
2. The transformation assumes all points have zero height, i.e. $Z = 0$. The arena where the tests where performed had a generally flat surface, but some bumps were present. See Section 6-1 for more details on the testing location.

3. The chosen mounting position of the camera introduces more uncertainty in the pixels higher in the frame (low $v$) than pixels lower in the frame (high $v$). In Figure 4-2 a grid of pixels with dimensions equal to the selected region of interest for tracking is transformed with the inverse homography matrix $\mathcal{H}^{-1}$. For better viewing purposes, the grid of pixels is downsampled by a factor five. The distance between subsequent pixels in the $X$-direction increases as the pixels are located higher in the frame, i.e. a smaller $v$-coordinate. The distance between pixels projections in the $Y$-direction increases also if $v$ increases, but does not depend on the $u$-coordinate.

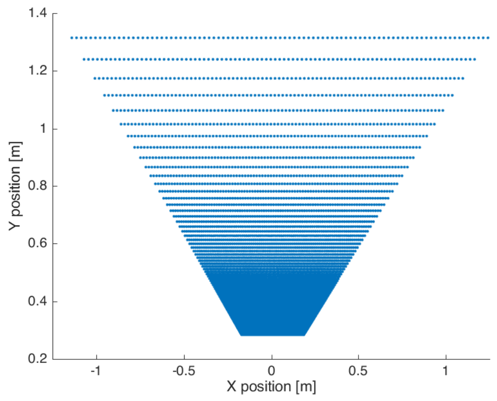
Using the observation that the distance to the next pixel depends on the pixel location, it is assumed that the transformation error also depends on the pixel location. The error on the x- and y-coordinate, respectively $(\epsilon_x(u_{k-1}, v_{k-1}), \epsilon_y(u_{k-1}, v_{k-1}))$ is therefore a function of the pixel position $p_k(u_{k-1}, v_{k-1})$. The functions $\epsilon_x, \epsilon_y$ give an approximation of the homography transformation error based on the location of a pixel. In section Section 4-4 a function will be created that can model this relationship. The equation for the disturbed real world point $P_{k-1}(\tilde{x}, \tilde{y})$ corresponding to $p_{k-1}(u_{k-1}, v_{k-1})$ is:

$$P_{k-1}(\tilde{x}_{k-1}, \tilde{y}_{k-1}) = \begin{bmatrix} \tilde{x}_{k-1} \\ \tilde{y}_{k-1} \end{bmatrix} = \underbrace{\begin{bmatrix} h_x^{-1}(u_{k-1}, v_{k-1}) \\ h_y^{-1}(u_{k-1}, v_{k-1}) \end{bmatrix}}_{P_{k-1}} + \underbrace{\begin{bmatrix} \epsilon_x(u_{k-1}, v_{k-1}) \\ \epsilon_y(u_{k-1}, v_{k-1}) \end{bmatrix}}_{\epsilon_{k-1}} \tag{4-1}$$

$$\tilde{P}_{k-1} = P_{k-1} + \epsilon_{k-1} \tag{4-2}$$



**(a)** Points in pixel coordinates



**(b)** Points in world coordinates

**Figure 4-2:** Grid of points describing the region of interest (left) converted to real world coordinates (right) using the homography transformation.

Where $h_i^{-1}(u, v), i = \{x, y\}$ represents the inverse homography transformation for each respective axis. The disturbed signal $P_{k-1}(\tilde{x}, \tilde{y})$ can then be split in a true signal and an error signal, as is done in (4-2). This type of error, the homography error, is also present when transforming the tracked pixel location to real world coordinates. In the next section it will be combined with the tracking error to define the total error on the real world position of the tracked pixel.

### 4-2-2 Tracking error

The second type of error that is added to the estimate is coming from the tracking algorithm. The Lucas-Kanade algorithm is not based on feature matching, but on predicting the place of a point in the next frame by solving a least squares problem based on image gradients. This leaves room for introducing an error. The pixel point $p_{k-1}(u_{k-1}, v_{k-1})$ will not only be moved by the true displacement $(\Delta u, \Delta v)$ but also a tracking error $(\epsilon_{\Delta u}, \epsilon_{\Delta v})$ resulting in a disturbed point $p_k(\tilde{u}_k, \tilde{v}_k)$. The disturbed coordinates $(\tilde{u}, \tilde{v})$ are defined as:

$$\tilde{u}_k = u_{k-1} + \Delta u + \epsilon_{\Delta u} \quad \tilde{v}_k = v_{k-1} + \Delta v + \epsilon_{\Delta v} \tag{4-3}$$

Transforming to real world coordinates is the third source of error that is considered for this project. The transformation again introduces a homography transformation error $(\epsilon_x, \epsilon_y)$. Now also the tracking error $(\epsilon_{\Delta u}, \epsilon_{\Delta v})$ propagates through the homography. Combining all this, the second world point $P_k(\tilde{x}_k, \tilde{y}_k)$ can be formulated:

$$P_k(\tilde{x}_k, \tilde{y}_k) = \begin{bmatrix} h_x^{-1}(\tilde{u}_k, \tilde{v}_k) + \epsilon_x(\tilde{u}_k, \tilde{v}_k) \\ h_y^{-1}(\tilde{u}_k, \tilde{v}_k) + \epsilon_y(\tilde{u}_k, \tilde{v}_k) \end{bmatrix} \tag{4-4}$$

The error of the tracking algorithm $(\epsilon_{\Delta u}, \epsilon_{\Delta v})$ will be propagated through the homography to the real world coordinate. An error of just a couple pixels on a point that is located in the top region of the frame (low $v$ coordinate) has a much bigger effect than that same error

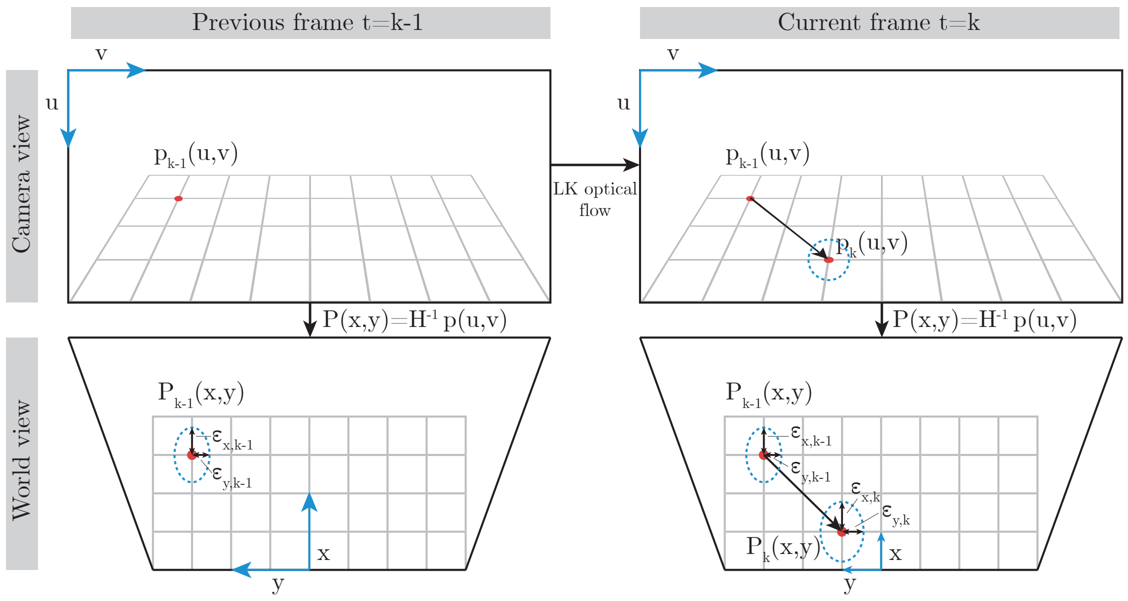


**Figure 4-3:** Overview of the points where error is injected into the system. Tracking introduces and error in the predicted point $p_k(u, v)$ and transformation from pixel coordinates to world coordinates introduces an error for both time steps.

located in the bottom region of the frame. To be able to split the disturbed world point $\tilde{P}_k$ into a true part and an error part, the assumption was made that the total error on $\tilde{P}_k$ consists of the homography transformation error based on the disturbed coordinates $(\tilde{u}_k, \tilde{v}_k)$ plus the effect of a tracking error corrected for the point position $p_k(\tilde{u}_k, \tilde{v}_k)$. Both errors will be approximated using heuristic functions. These functions will depend respectively on the pixel location for the transformation error and the corner quality $q_{p_{k-1}}$, the aforementioned forward-backward projection error $d_{p_k}$ and also the pixel location for the tracking error. All this and the chosen variables to base the functions on will later be discussed in Section 4-4. Adopting this split in two separate, uncoupled errors allows for $P_k(\tilde{x}_k, \tilde{y}_k)$ to be written as:

$$P_k(\tilde{x}_k, \tilde{y}_k) \approx \underbrace{\begin{bmatrix} h_x^{-1}(u_{k-1} + \Delta u, v_{k-1} + \Delta v) \\ h_y^{-1}(u_{k-1} + \Delta u, v_{k-1} + \Delta v) \end{bmatrix}}_{P_k} + \underbrace{\begin{bmatrix} \epsilon_{x,h}(\tilde{u}_k, \tilde{v}_k) + \epsilon_{\Delta u}(\tilde{u}_k, \tilde{v}_k, q_{p_{k-1}}, d_{p_k}) \\ \epsilon_{y,h}(\tilde{u}_k, \tilde{v}_k) + \epsilon_{\Delta v}(\tilde{u}_k, \tilde{v}_k, q_{p_{k-1}}, d_{p_k}) \end{bmatrix}}_{\epsilon_k} \tag{4-5}$$

For the point at sample $k - 1$ one error is modelled, the homography error based on the detected point and for the tracked point to $k$ two errors are modelled, the tracking error and the homography error. The whole process and the three moments where error is introduced can also be seen in Figure 4-3 in a more visual way. These three sources will not be a full reconstruction of all the possible sources of error, but should give enough degrees of freedom to design a good filter based on the covariance estimation. Later it must be validated that these sources were sufficient to create a Dynamic Covariance Estimation algorithm.

## 4-3 Measurement covariance submatrix

Both the previous and the current point can now be split into a true signal and an error signal. For the error signal an approximation was adopted in order to model which errors make up the error signal and on which parameters they depend. The final goal of this chapter is to model the variance of the egomotion $V_x, V_y$ according to one point pair $p_{k-1}(u_{k-1}, v_{k-1}), p_k(u_k, v_k)$. To do this, the variance equations for the egomotion first need to be established. A first step in doing this, is the process model with process noise and measurement noise from Section 2-3:

$$\mathbf{x}_k = A_k \mathbf{x}_{k-1} + B\mathbf{u}_k + \mathbf{w}_k \tag{4-6}$$

$$\mathbf{y}_k = h_k(\mathbf{x}_k) + \mathbf{v}_k \tag{4-7}$$

The process noise $\mathbf{w}_k$ is an unknown random process representing unmodelled error in the process. The measurement noise $\mathbf{v}_k$ represents the unmodelled error in the sensors creating the measurements of the system states. The covariance matrix for this system model is defined as:

$$E \begin{bmatrix} w(k) \\ v(k) \end{bmatrix} \begin{bmatrix} w(j)^T & v(j)^T \end{bmatrix} = \begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \Delta(k-j) \tag{4-8}$$

The covariance matrix consists of four different submatrices. Submatrix $Q$ defines the covariance of the process noise $\mathbf{w}_k$. With the chosen kinematic model as a basis, this can be attributed to noise on the acceleration and yaw rate sensor and is assumed to be constant. For now it assumed that there is no correlation between the process noise on $V_x$ and $V_y$ to simplify tuning of the observer in a later stage. This will make the $Q$ matrix diagonal. Next to that the assumption is also made that there is no cross-correlation between the process noise and the measurement noise, so $S$ is equal to zero. The only submatrix that will be adjusted to make better use of the camera measurements is the measurement noise covariance submatrix $R$.

The goal of this chapter is to create functions that can adjust covariance submatrix R, based on a model of the variance of one calculated velocity (egomotion). Consequently it is useful to write out the variance equation for the egomotion for one tracked point between two frames. As an example the longitudinal velocity $V_{x,k}$ for a point $p$ at sample $k$ is chosen. Later in the EKF, the measurements will be based on not just one point $p$ but all points $[p_1 \dots p_{N_p}]$ tracked in one frame. The calculated velocity of the vehicle based on a tracked point is defined as:

$$V_{x,k} = \frac{p_{x,k} - p_{x,k-1}}{\Delta t} + p_{y,k-1}\omega_{z,k} \tag{4-9}$$

The disturbed velocity $\tilde{V}_{x,k}$ can be written as a calculation based on disturbed distances and angular velocity:

$$\tilde{V}_{x,k} = \frac{\tilde{p}_{x,k} - \tilde{p}_{x,k-1}}{\Delta t} + \tilde{p}_{y,k-1}\tilde{\omega}_{z,k} \tag{4-10}$$

Combining this equation with the assumed error sources from Section 4-2 and assuming the yaw rate contains a time independent error $\epsilon_{\omega,z}$ resulting in $\tilde{\omega}_{z,k} = \omega_{z,k} + \epsilon_{\omega,z}$, (4-10) can be

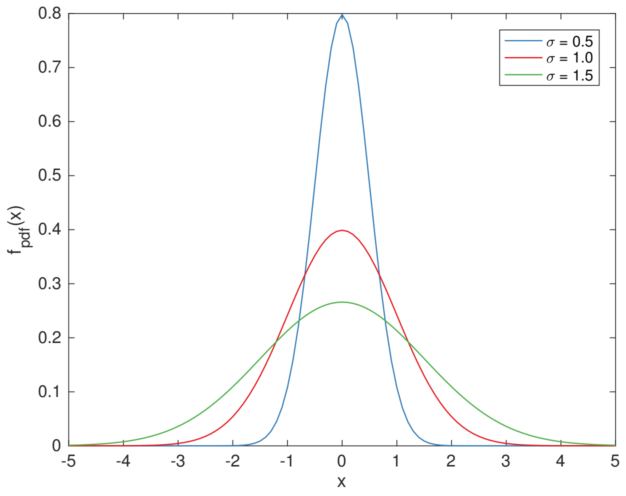


**Figure 4-4:** Gaussian probability density function for three values of $\sigma$

rewritten as:

$$\tilde{V}_{x,k} = \frac{(p_{x,k} + \epsilon_{x,k}) - (p_{x,k-1} + \epsilon_{x,k-1})}{dt} + (p_{y,k-1} + \epsilon_{y,k-1})(\omega_{z,k} + \epsilon_{\omega,k}) \tag{4-11}$$

$$= \underbrace{\frac{p_{x,k} - p_{x,k-1}}{dt} + p_{y,k-1}\omega_{z,k}}_{V_{x,k}} + \underbrace{\frac{\epsilon_{x,k} - \epsilon_{x,k-1}}{dt} + \epsilon_{y,k-1}\epsilon_{\omega,k} + \epsilon_{y,k-1}\omega_{z,k} + p_{y,k-1}\epsilon_{\omega,k}}_{v(k)} \tag{4-12}$$

$$v(k) = \frac{\epsilon_{x,k} - \epsilon_{x,k-1}}{dt} + \epsilon_{y,k-1}\epsilon_{\omega,k} + \epsilon_{y,k-1}\omega_{z,k} + p_{y,k-1}\epsilon_{\omega,k} \tag{4-13}$$

$$\tag{4-14}$$

The signal $v(k)$ now models the measurement error signal of one single calculated longitudinal velocity at time $k$. It contains different error terms $\epsilon$. Up until this point, it was assumed that those error terms were custom functions based on certain variables. To get a framework to create these, each error is modelled as a separate, uncorrelated, random Gaussian process with a variable variance. This means each $\epsilon$ has zero mean, $E[\epsilon] = 0$, a variance $\sigma^2$, $E[\epsilon\epsilon^T] = \sigma^2$, and the Probability Density Function (PDF) is shaped like a Gaussian distribution. The Gaussian distribution is defined as:

$$f_{pdf}(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \tag{4-15}$$

Since the error is zero mean, the shape of the PDF only depends on $\sigma$. In Figure 4-4 the probability density function for several variations of $\sigma$ can be seen. By scaling the variance $\sigma^2$ the uncertainty on some measurement can be increased or decreased based on chosen parameters.

To adjust the variance on camera measurements, the base variance $\sigma^2$ will be scaled by some arbitrary function $f(x)$, that depends on the variables that are relevant for that error, resulting in the scaled variance $\bar{\sigma}^2$:

$$\bar{\sigma}^2 = \sigma^2_{\text{base}} f(x) \tag{4-16}$$

In this way, the variance of the error increases if the scaling function $f(x)$ increases, but the other properties of the random noise process, mainly the zero mean and uncorrelated property are (assumed to be) preserved. Although it is not known if the selected errors exactly behave like random Gaussian processes, assuming it is the case enables writing the variance of $v(k)$ in an elegant and compact way:

$$E[v(k)v(k)^T] = E[\frac{\epsilon^2_{x,k-1}}{dt^2} + \frac{\epsilon^2_{x,k}}{dt^2} + \epsilon^2_{y,k-1}\epsilon^2_{\omega,k} + \epsilon^2_{\omega,k}p^2_{y,k-1} + \epsilon^2_{y,k-1}\omega^2_{z,k} + \text{cross terms}] \tag{4-17}$$

$$= E\left[\frac{\epsilon^2_{x,k-1}}{dt^2} + \frac{\epsilon^2_{x,k}}{dt^2} + \epsilon^2_{y,k-1}\epsilon^2_{\omega,k} + \epsilon^2_{\omega,k}p^2_{y,k-1} + \epsilon^2_{y,k-1}\omega^2_{z,k}\right] \tag{4-18}$$

$$= \frac{\sigma^2_{x,k-1} + \sigma^2_{x,k}}{dt^2} + \sigma^2_{y,k}\sigma^2_{\omega,k} + \sigma^2_{\omega,k}p^2_{y,k} + \sigma^2_{y,k}\omega^2_{z,k} \tag{4-19}$$

All cross terms, the terms that contain at least one $\epsilon$ that is not squared, are cancelled out under the assumption that error sources have zero mean and are uncorrelated. Only some variance terms, the lateral point position and the yaw rate then remain in (4-19). The final assumption made in the process of modelling the covariance submatrix $R$ is that there is no cross-correlation between points, hence $R$ is diagonal. Later it has to be verified if this simplification gives adequate results. This will make the submatrix $R$ diagonal. For the full mathematics to go from (4-13) to (4-18) see Appendix B-3. The variance function above can also be derived for the lateral speed in an analogous way, yielding:

$$E[v(k)v(k)^T] = \frac{\sigma^2_{y,k} + \sigma^2_{y,k-1}}{dt^2} + \sigma^2_{x,k-1}\sigma^2_{\omega,k} + \sigma^2_{\omega,k}p^2_{x,k-1} + \sigma^2_{x,k-1}\omega^2_{z,k} \tag{4-20}$$

It can be seen that the variance for the calculated speed in one direction does not only depend on the variance of the tracked points in world coordinates in that respective direction, but the yaw rate correction also has a big impact. Correcting for the yaw rate of the vehicle propagates both the point variance in the other direction, the variance of yaw rate and the value of those two quantities *itself* into the covariance. This is an important observation for creating the covariance functions later.

## 4-4 Variance scaling functions

In the previous section equations to approximate the measurement covariance of the vehicle speed based on camera measurements were created. In fact, it only consisted of a variance, since there is no covariance between signals. The next step is to use these equations to improve the observer by adjusting the measurement covariance submatrix during operation to the optimal value. This is done by scaling the various $\sigma$ terms in (4-19) and (4-20). The scaling of these terms is done using heuristic functions, based on the input variables established in Section 4-2. These scaled $\sigma$ terms were plugged into the variance equations

(4-19) to (4-20). Equations to model the measurement noise variance of respectively $V_{i,x,k}$ and $V_{i,y,k}$, the velocity based on the displacement of point pair number $i$ along axis $x$ or $y$ from sample $k-1$ to $k$ will now have the following structure:

$$r_{i,V_x,k} = \frac{\bar{\sigma}^2_{i,x,k-1}}{dt^2} + \frac{\bar{\sigma}^2_{i,x,k}}{dt^2} + \bar{\sigma}^2_{i,y,k}\sigma^2_{\omega,k} + \sigma^2_{\omega,k}p^2_{i,y,k} + \bar{\sigma}^2_{i,y,k}\omega^2_{z,k} \tag{4-21}$$

$$r_{i,V_y,k} = \frac{\bar{\sigma}^2_{i,y,k-1}}{dt^2} + \frac{\bar{\sigma}^2_{i,y,k}}{dt^2} + \bar{\sigma}^2_{i,x,k}\sigma^2_{\omega,k} + \sigma^2_{\omega,k}p^2_{i,x,k} + \bar{\sigma}^2_{i,x,k}\omega^2_{z,k} \tag{4-22}$$

In Section 4-2 some assumptions for modelling these errors were already made. The scaled variances $\bar{\sigma}_{i,x,k-1}$ and $\bar{\sigma}_{i,y,k-1}$ will scale with the homography error at frame $k-1$, the other variances $\bar{\sigma}_{i,x,k}$ and $\bar{\sigma}_{i,y,k}$ will scale with both the homography error at frame $k$ and the tracking error from frame $k-1$ to frame $k$. Therefore first the scaling based on homography error will be modelled with a heuristic function and then the tracking error will be looked after.

## 4-4-1 Homography variance scaling

Scaling based on the homography error should model the increasing uncertainty in real world projection of pixels as the location of a pixel changes. The physical background used for the scaling in this project will be the incremental pixel distance. This constructed quantity describes the increase in distance along one of both real world axes (X or Y) if the pixel location were moved by one pixel along the corresponding image axis (resp. $v$ or $u$). This quantity will be called $\Delta_i(p(u,v))$ $i = \{x, y\}$:

$$\Delta_x(p(u,v)) = h_x^{-1}(u, v+1) - h_x^{-1}(u,v) \quad \Delta_y(p(u,v)) = h_y^{-1}(u+1, v) - h_y^{-1}((u,v)) \tag{4-23}$$

Where $h_x^{-1}, h_y^{-1}$ are functions transforming pixel coordinates to real world coordinates based on the homography matrix $\mathcal{H}$. As can also be seen in Figure 4-5, the distance between two neighboring pixels in real world coordinates depends on their location in pixel coordinates.

It can be seen that pixels with the same $u$ coordinate in the image lie on the same horizontal line in real world coordinates and pixels with the same $v$ coordinate lie on the same diagonal line. The second observation is that the distance to the next pixel along either the $x$- or $y$-axis in real world coordinates only depends on the $v$ coordinate. For the X-axis, the distance to the next pixel increases as the $v$ coordinate increases. For the Y-axis, the distance to the next pixel is equal for all pixels on the same horizontal line. This is a result of the camera only being tilted down and not being yawed.

The error that is introduced when transforming from pixel coordinates to real world coordinates is again assumed to behave like a random Gaussian process. This can be viewed as the possibility that a pixel is not always exactly transformed to its real world equivalent, but in a Gaussian shaped distribution around that point. The assumption made here is that the width of this shape, which is directly linked to $\sigma^2$, increases as the $v$ coordinate decreases. To model this behavior, the squared incremental pixel distance in both the X- and Y-direction

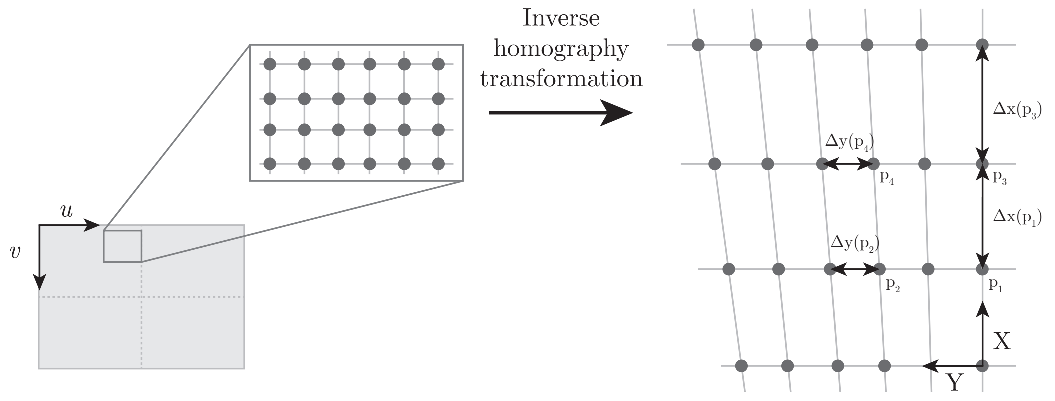


**Figure 4-5:** Visualization of the distance to the next pixel projection along the X- and Y-direction, where the points $p_1 - p_4$ represent pixels projected in the real world coordinate system.

for all pixels has been calculated using the calibrated homography and is plotted in Figures 4-6a to 4-6b. The squared distance is used since we want to let this function scale up the variance. Therefore the squared "error" is used here. As expected, the plane is only curved along the $v$-direction. To estimate the relationship between $v$ coordinate and squared error relationship, a vertical line was drawn in the frame at $u = 450$ and for every $v$ the incremental pixel distance was calculated. Then a line was fit through all the incremental pixel distance points. Results of this can be seen in Figures 4-6c to 4-6d. The fitted functions types are:

$$f_{h,x}(v) = \frac{p_1 v^2 + p_2 v + p_3}{v^4 + q_1 v^3 + q_2 v^2 + q_3 v + q_4} \qquad f_{h,y}(v) = \frac{p_1 v + p_2}{v^2 + q_2 v + q_2} \tag{4-24}$$

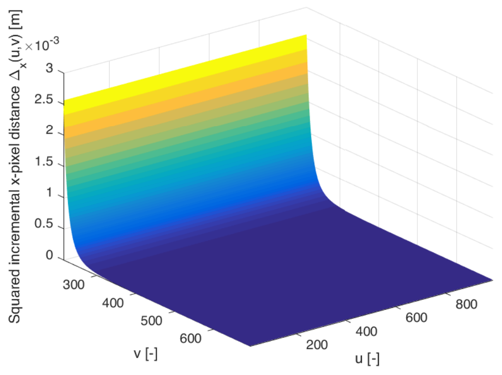
$$\bar{\sigma}^2_{x,k-1} = \sigma^2_{\text{hom}} f_{h,x}(v) \qquad \bar{\sigma}^2_{y,k-1} = \sigma^2_{\text{hom}} f_{h,y}(v) \tag{4-25}$$

$$\tag{4-26}$$

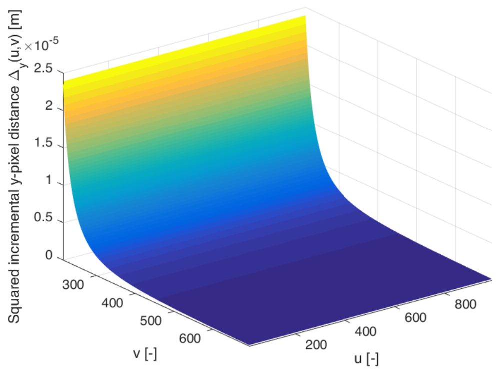
Where $f_{h,x}$ is the scaling function for the X-position variance and $f_{h,y}$ is the scaling function for the Y-position variance. Both functions scale the same base variance $\sigma^2_{\text{hom}}$. It can be seen that the scaling function for the X-distance was chosen as a fourth order rational function and the function for the Y-distance as a second order rational function. The functions were fit using a nonlinear least squares optimization using a trust-region algorithm and both curves had a Root Mean Square Error (RMSE) of $< 1 \cdot 10^{-6}$. The resulting parameters are listed in Table 4-1.

**Table 4-1:** The fitted parameters for the homography error scaling functions

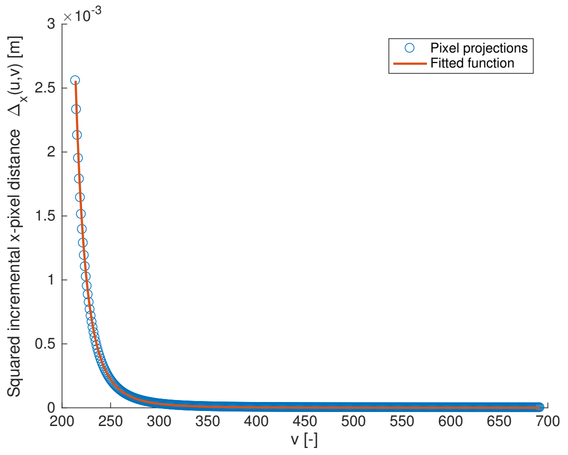
| X-axis error | | Y-axis error | |
|---|---|---|---|
| Parameter | Value | Parameter | Value |
| $p_1$ | $5.023 \times 10^{-2}$ | $p_1$ | $1.801 \times 10^{-4}$ |
| $p_2$ | $-5.119 \times 10^{1}$ | $p_2$ | $1.414 \times 10^{-1}$ |
| $p_3$ | $2.417 \times 10^{4}$ | | |
| $q_1$ | $-5.698 \times 10^{2}$ | $q_1$ | $-1.945 \times 10^{2}$ |
| $q_2$ | $1.097 \times 10^{5}$ | $q_2$ | $6.325 \times 10^{-1}$ |
| $q_3$ | $-7.323 \times 10^{6}$ | | |
| $q_4$ | $3.535 \times 10^{7}$ | | |



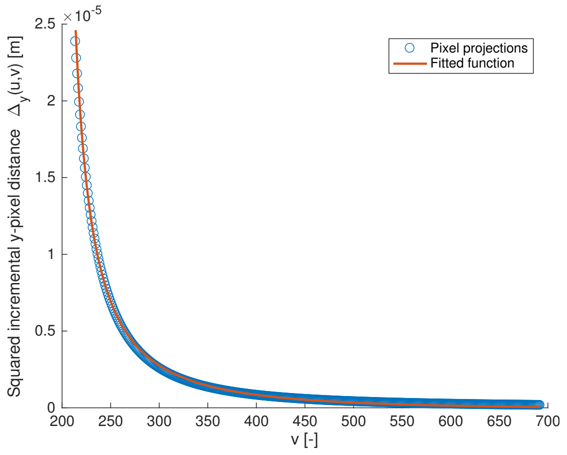
**(a)** Distance to next pixel in X-direction for all points



**(b)** Distance to next pixel in Y-direction for all points



**(c)** Distance to next pixel in X-direction for $u = 450$ for all pixels together with fitted function



**(d)** Distance to next pixel in Y-direction for $u = 450$ for all pixels together with fitted function

**Figure 4-6:** Overview of distance to next pixel projection in X- and Y-direction for all points (top) and then for a vertical line with $u = 450$.

### 4-4-2 Tracking variance scaling

The second pair of variances that need to be scaled are $\sigma^2_{x,k}$ and $\sigma^2_{y,k}$. As said before, scaling of the variance of the second world point is assumed to depend on two separate error signals. The first being the tracking error and the second again the homography error. The homography error is modelled in exactly the same way as described in the previous section using (4-24), only now the disturbed coordinates $(\tilde{u}_k, \tilde{v}_k)$ will be entered instead of the coordinates before tracking $(u_{k-1}, v_{k-1})$. These disturbed coordinates are the coordinates coming out of the Iterative Pyramidal Lucas Kanade Optical Flow (IPLKOF) tracking algorithm.

For scaling according to the tracking quality two different properties of a point will be used. The first is the corner quality, the second is the forward-backward projection error. A corner with a high corner quality, which is the minimum eigenvalue of the corner matrix at that point, indicates that a strong change in brightness is available in both directions. It was expected that this should make it easier to track that corner, thus resulting in a lower variance of the tracking error. The forward-backward error was used in [58] to detect outliers in tracking. For the variance scaling it was assumed that a larger forward-backward error should indicate a larger variance of the tracking error. Next to these two quantities, the position of the point in pixel coordinates should also be taken into account. A tracking error of 10 pixels will have a much greater influence in the top of the frame than in the bottom of the frame once converted to real world coordinates. Therefore the normalized point quality and normalized forward-backward error are defined as:

$$q'_{p_{k-1}} = \frac{q_{p_{k-1}}}{\sqrt{f^2_{h,x}(v) + f^2_{h,y}(v)}} \tag{4-27}$$

$$d'_{p_k} = d_{p_{k-1}} \sqrt{f^2_{h,x}(v) + f^2_{h,y}(v)} \tag{4-28}$$

Here $q'_{p_{k-1}}$ represents the normalized point quality and $d'_{p_k}$ the normalized forward-backward error. Because $q_{p_k}$ is a quality, e.g. a high value should indicate good tracking, it is divided by the norm of the homography scaling functions evaluated at that point to obtain $q'_{p_k}$. The adjusted forward-backward error $d'_{p_k}$ is created by multiplying the normal forward-backward error $d_{p_k}$ by the norm of both homography scaling functions. In that way an error at a point with a high $v$ coordinate (bottom of the frame) is penalized less then the same error at a low $v$ coordinate (top of the frame).

After this, scatter plots were made of the transformed corner quality and forward-backward error of every tracked point versus the squared error between the lateral velocity according to that point (based on egomotion calculation) and the true Motion Capture System velocity. This was done on a maneuver with a constant torque of 1.75 N and a ramp steering input from 7° to 16°, maneuver 4A (see Table 6-1). A longitudinal speed of $1.75\,\mathrm{m\,s^{-1}}$ and a lateral speed of 0.1-0.2 $\mathrm{m\,s^{-1}}$ were achieved. These scatter plots can be seen in Figure 4-7. Next to error points, a histogram of 50 bins with error bars of $\pm\sigma$ is plotted. The average error per bin is solely used for viewing purposes, because it is hard to see trends in the plot with $2.34 \times 10^5$ points, not for fitting purposes. Finally the fitted scaling functions are also plotted in Figure 4-7. The fitted scaling functions are:

$$f_{t,1}(q'_{p_{k-1}}) = a_1 {q'}^{\,b_1}_{p_{k-1}} \quad f_{t,2}(d'_{p_k}) = a_1 {d'}^{\,b_1}_{p_k} \tag{4-29}$$

**Table 4-2:** The fitted parameters for the homography error scaling functions

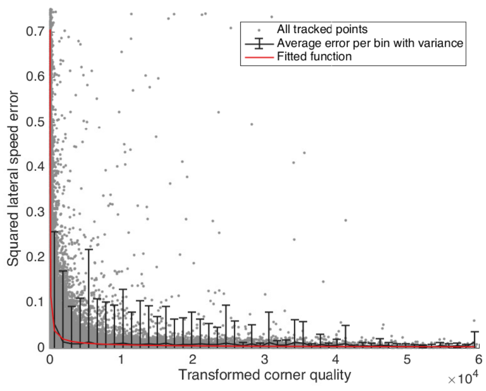
| **Corner quality scaling** | | **FB-error scaling** | |
|---|---|---|---|
| Parameter | Value | Parameter | Value |
| $a_1$ | 1.88 | $a_1$ | 20.64 |
| $b_1$ | $-0.6094$ | $b_1$ | 0.5135 |

Where $a$ and $b$ are parameters that need to be estimated. The scaling functions based on corner quality and forward-backward error were again fitted with a nonlinear least squares optimization using a Levenberg-Marquadt algorithm on all data points. The estimated parameters can be seen in Table 4-2. The RMSE for the corner quality function equalled 0.2113, RMSE for the forward backward error equalled 0.2112. A single power function was chosen to prevent overfitting of the data in the regions were few point are available. The final scaled variances $\bar{\sigma}^2_{x,k}$ and $\bar{\sigma}^2_{y,k}$ can now be formulated. They were assumed to consist of the homography error and the tracking error. Therefore the scaling of the variance of the Gaussian noise error signal will be done by the homography scaling function and an affine sum of both the tracking-error based scaling functions.

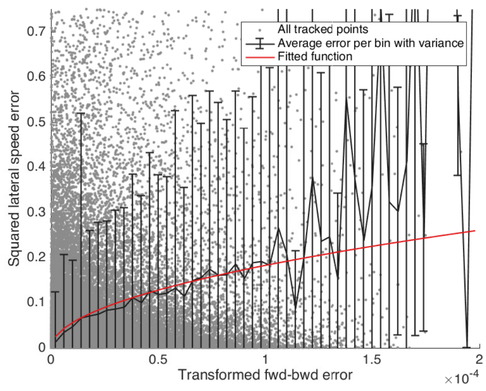
$$\bar{\sigma}^2_{x,k} = \sigma^2_{\text{hom}} f_{h,x}(v) + \sigma^2_{\text{track}} \left( \alpha f_{t,1}(q'_{p_{k-1}}) + (1-\alpha) f_{t,2}(d'_{p_k}) \right) \tag{4-30}$$

$$\bar{\sigma}^2_{y,k} = \sigma^2_{\text{hom}} f_{h,y}(v) + \sigma^2_{\text{track}} \left( \alpha f_{t,1}(q'_{p_{k-1}}) + (1-\alpha) f_{t,2}(d'_{p_k}) \right) \tag{4-31}$$

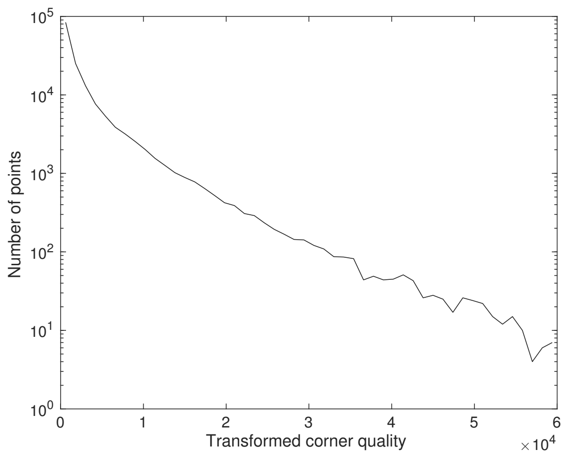
The base covariance $\sigma^2_{\text{hom}}$ is the same as was used for scaling $\bar{\sigma}^2_{x,k-1}$ and $\bar{\sigma}^2_{y,k-1}$ and $\sigma^2_{\text{track}}$ has the same value for both equations. Parameter $\alpha$ determines in which ratio both tracking scaling functions are summed.



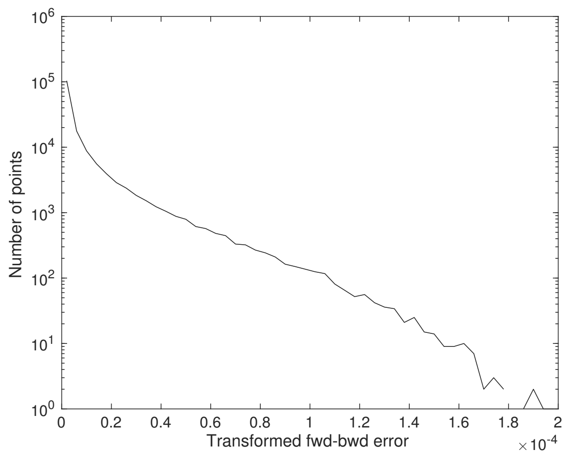
**(a)** Absolute speed error of all tracked points vs transformed corner quality, average error per bin with symmetric error bars of $\pm\sigma$ and fitted scaling function



**(b)** Absolute speed error of all tracked points vs transformed forward-backward error, average error per bin with symmetric error bars of $\pm\sigma$ and fitted scaling function



**(c)** Number of points per bin vs transformed corner quality



**(d)** Number of points per bin vs transformed forward backward error

**Figure 4-7:** Absolute speed error vs transformed corner quality (left) and forward-backward error (right) for a constant torque, $1.75\,\mathrm{m\,s^{-1}}$, $5°$-$20°$ ramp steer. Number of bins $= 50$, bin size for corner quality $= 1.2 \times 10^3$, bin size for fwd-bwd error $= 4.0 \times 10^{-6}$. Total data points $2.34 \times 10^5$.

## 4-5 DCE overview

After establishing a model for the variance of one calculated velocity and fitting functions that can scale this variance, the DCE is complete. The covariance submatrices $Q$ and $R$ at sample $k$ can now be calculated using the DCE:

$$Q = \begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix} \qquad R_k = \begin{bmatrix} r_{enc} & 0 & 0 & 0 \\ 0 & R_{V_{x,k}} & 0 & 0 \\ 0 & 0 & R_{V_{y,k}} & 0 \\ 0 & 0 & 0 & r_{a_y} \end{bmatrix} \tag{4-32}$$

$$R_{V_{x,k}} = \begin{bmatrix} r_{1,V_x,k} & 0 & \cdots & 0 \\ 0 & r_{2,V_x,k} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{N_p,V_x,k} \end{bmatrix} R_{V_{y,k}} = \begin{bmatrix} r_{1,V_y,k} & 0 & \cdots & 0 \\ 0 & r_{2,V_y,k} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{N_p,V_y,k} \end{bmatrix} \tag{4-33}$$

The submatrices $R_{V_{x,k}}$ and $R_{V_{y,k}}$ are diagonal and have dimensions $\mathbb{R}^{N_p \times N_p}$. This results in an $R_k$ of size $\mathbb{R}^{2+2N_p \times 2+2N_p}$. Each individual diagonal entry of the matrices $R_{V_{x,k}}$ and $R_{V_{y,k}}$ is calculated using:

$$r_{i,V_x,k} = \frac{\bar{\sigma}^2_{i,x,k-1}}{dt^2} + \frac{\bar{\sigma}^2_{i,x,k}}{dt^2} + \bar{\sigma}^2_{i,y,k}\sigma^2_{\omega,k} + \sigma^2_{\omega,k}p^2_{i,y,k} + \bar{\sigma}^2_{i,y,k}\omega^2_{z,k} \tag{4-34}$$

$$r_{i,V_y,k} = \frac{\bar{\sigma}^2_{i,y,k-1}}{dt^2} + \frac{\bar{\sigma}^2_{i,y,k}}{dt^2} + \bar{\sigma}^2_{i,x,k}\sigma^2_{\omega,k} + \sigma^2_{\omega,k}p^2_{i,x,k} + \bar{\sigma}^2_{i,x,k}\omega^2_{z,k} \tag{4-35}$$

$$\bar{\sigma}^2_{x,k-1} = \sigma^2_{\text{hom}} f_{h,x}(v) \tag{4-36}$$

$$\bar{\sigma}^2_{y,k-1} = \sigma^2_{\text{hom}} f_{h,y}(v) \tag{4-37}$$

$$\bar{\sigma}^2_{x,k} = \sigma^2_{\text{hom}} f_{h,x}(v) + \sigma^2_{\text{track}}\left(\alpha f_{t,1}(q'_{p_{k-1}}) + (1-\alpha) f_{t,2(}d'_{p_k})\right) \tag{4-38}$$

$$\bar{\sigma}^2_{y,k} = \sigma^2_{\text{hom}} f_{h,y}(v) + \sigma^2_{\text{track}}\left(\alpha f_{t,1}(q'_{p_{k-1}}) + (1-\alpha) f_{t,2(}d'_{p_k})\right) \tag{4-39}$$

$$q'_{p_{k-1}} = \frac{q_{p_{k-1}}}{\sqrt{f^2_{h,x}(v) + f^2_{h,y}(v)}} \tag{4-40}$$

$$d'_{p_k} = d_{p_k}\sqrt{f^2_{h,x}(v) + f^2_{h,y}(v)} \tag{4-41}$$

The four scaling functions of the DCE, $f_{h,x}(v), f_{h,y}(v), f_{t,1}(q'_{p_{k-1}})$ and $f_{t,2}(d'_{p_k})$, have been established in the previous sections. Only four parameter remain for tuning the DCE: $\sigma^2_{\text{hom}}, \sigma^2_{\text{track}}, \alpha$ and $\sigma^2_{\omega,z}$. These parameters will be tuned in Section 6-2.

## 4-6 Conclusion

All subparts for scaling the variance of one measurement are now known. The $\bar{\sigma}^2_{x,k-1}, \bar{\sigma}^2_{y,k-1}$ variances can be scaled using a scaling function based on the used homography matrix, the $\bar{\sigma}^2_{x,k}, \bar{\sigma}^2_{y,k}$ can be scaled using both the homography based scaling function and a corner quality and tracking error based scaling function. These four scaled variances will then be used in the measurement covariance submatrix from the observer to increase the noise variance for some measurements more then for other measurements. This should obtain superior filtering when compared to using a static covariance matrix where for all camera measurements the same noise variance is used.

# Chapter 5

# Implementation

A full methodology for fusing Inertial Measurement Unit (IMU), encoder and camera measurements has now been proposed. In this chapter the implementation of the methodology will be discussed, in order to test the proposed method. First, some extra parts of the observer that are needed for good operation are discussed together with a full overview of the proposed observer. Next, the Motion Capture System (MCS) is discussed. This is the system that delivers the validation data for the system identification. The identification of the necessary vehicle parameters is finally presented in the last section, which consists of the tire force functions and steering law.

## 5-1 BARC Platform

The developed estimation methodology will be tested on the 1:10 scale remotely operated BARC vehicle. The BARC features all the sensors that have been discussed so far for the observer methodology. In this section the simulation setup that was used for testing the observer will be discussed. The sensors of the Berkeley Autonomous Race Car (BARC) have the following characteristics:

- IMU: $a_x, a_y, a_z$ $\omega_w, \omega_y, \omega_z$[1] at 100 Hz
- Encoder: $V_{enc,F}$ at 100 Hz
- Camera: 30 frames per second (fps) of at a resolution of 1024×768 pixels

The way the sensors are interconnected can be seen in Figure 5-1. The Odroid XU4 is the heart of the BARC, which connects either directly with sensors over USB, or indirectly via an Arduino Nano. The Odroid XU4 runs Lubuntu 14.04 with Robot Operating System (ROS) Indigo which is responsible for executing the preprogrammed open-loop maneuvers

[1]The IMU also features a three degrees of freedom magnetometer, but these measurements were not used in this project.

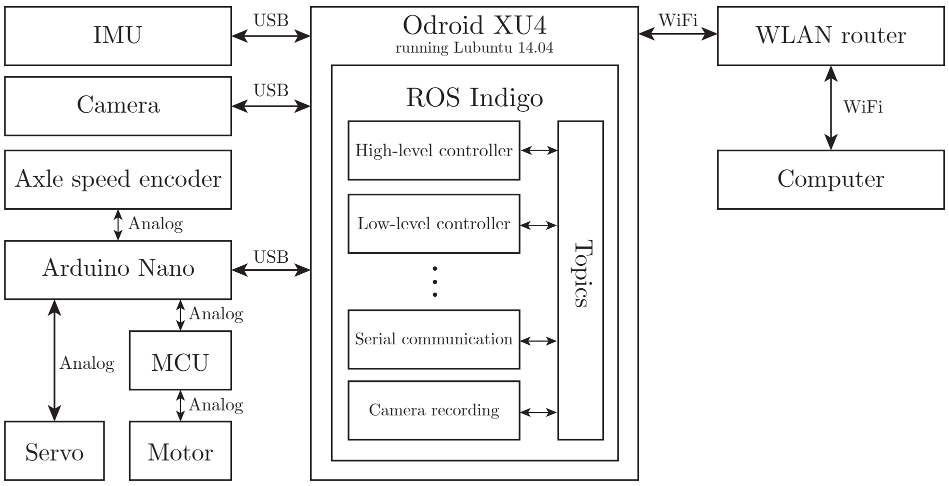


**Figure 5-1:** Block diagram of all the components of the BARC and the way they are linked up. The software overview of the Odroid XU4 is a simplified version. During operation more nodes are present.

and recording all sensor data. In the ROS architecture several nodes run in parallel, each responsible for a separate task. An overview of all nodes and their functions can be found in Appendix A-1.

The IMU for this project runs at 100 Hz and passes unfiltered signals over USB to the Odroid XU4. The IMU can already filter measurements internally, but this feature was disabled for the project. The Arduino Nano gives an update on the most recent time between a change in signal coming from the encoder at a rate of 100 Hz. Based on the time between two changes in signal, the time for one quarter rotation of the wheel and so the wheel speed can be calculated. This methodology was laid out in Subsection 2-2-2. The only extra processing done to the proposed methodology was applying a moving average filter of the past 5 sampels. The camera signal gets captured by a node running within the ROS architecture at a rate of 30 Hz and a resolution of $1024 \times 768$ pixels. The camera can also output at other resolutions, but obtaining a higher resolution comes at the cost of a lower frame rate.

The estimation algorithm was simulated offline as if it would run online. The test vehicle performed a preprogrammed open loop maneuver, see Section 6-1 for all used maneuvers. All signals from the vehicle were recorded and taken to a laptop to simulate the observer in Matlab. Signals with a frequency higher than 30 Hz where downsampled and interpolated to match the 30 Hz camera signal and obtain synchronous measurements. Then the observer including the Computer Vision Algorithm (CVA) and Dynamic Covariance Estimation (DCE) was run.

No tools or functions were used that wouldn't be available when running the algorithm online with all sensors running synchronous at 30 Hz. The only difference is that calculation time from the CVA is not factored in. Tracking data from frame $k$ is available at frame $k$, whereas online this could take up to one sample $T_s$ to calculate [44]. In the next two sections the final auxiliary elements of the observer are discussed to make the observer a success in the chosen simulation setup and after that the system identification of the BARC is presented.

## 5-2 Observer overview

In this section an overview of the developed observer will be given. This will combine information from the previous three chapters into one comprehensible overview. Before the full observer overview can be given, the signal preprocessing is presented first.

### 5-2-1 Signal preprocessing

Signals from the encoder and IMU are collected at a sample rate higher than 30 Hz. To be able to combine these signals with the camera measurements, the signals are downsampled. To prevent aliasing a low-pass filter at half the camera sampling frequency was implemented. The implementation used is a 2nd order Infinite Impulse Response Butterworth filter with $f_c = 15\,\text{Hz}$. This filter was chosen because of its gentle phase delay at lower frequencies. The magnitude and phase response of the filter can be seen in Figure 5-2. After filtering, the signals are downsampled and interpolated, to simulate sensors running synchronous with the camera.

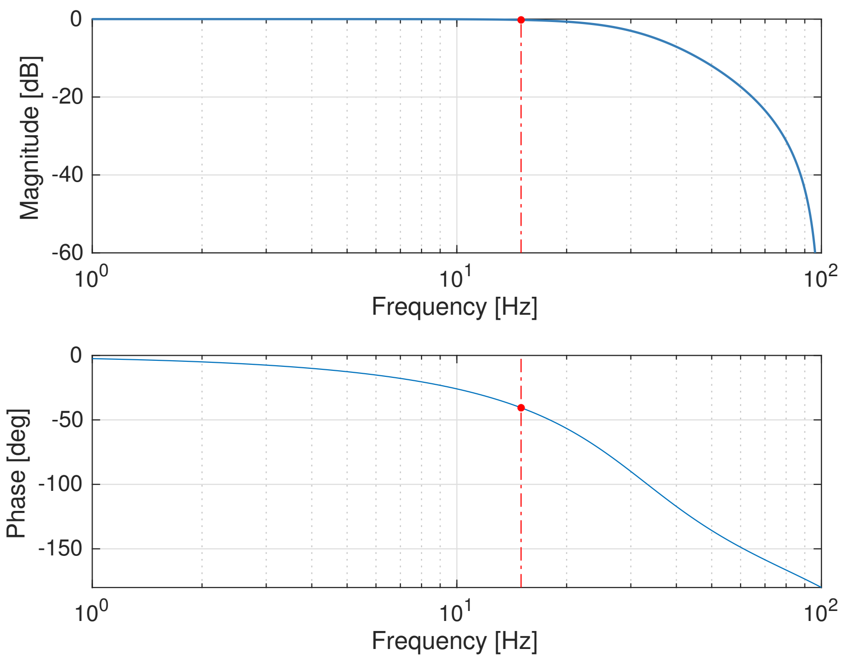


**Figure 5-2:** Mangitude and phase response of the anti-aliasing filter. Cut off frequency is marked with a red dot and located at $f_c = 15\,\text{Hz}$.

### 5-2-2 Full observer overview

The observer from Chapter 2 can now be combined with the CVA, the dynamic covariance estimation and the extra parts discussed in this chapter to form the full observer that was tested for this project. The overview of all the different subparts of the observer can be seen in Figure 5-3.

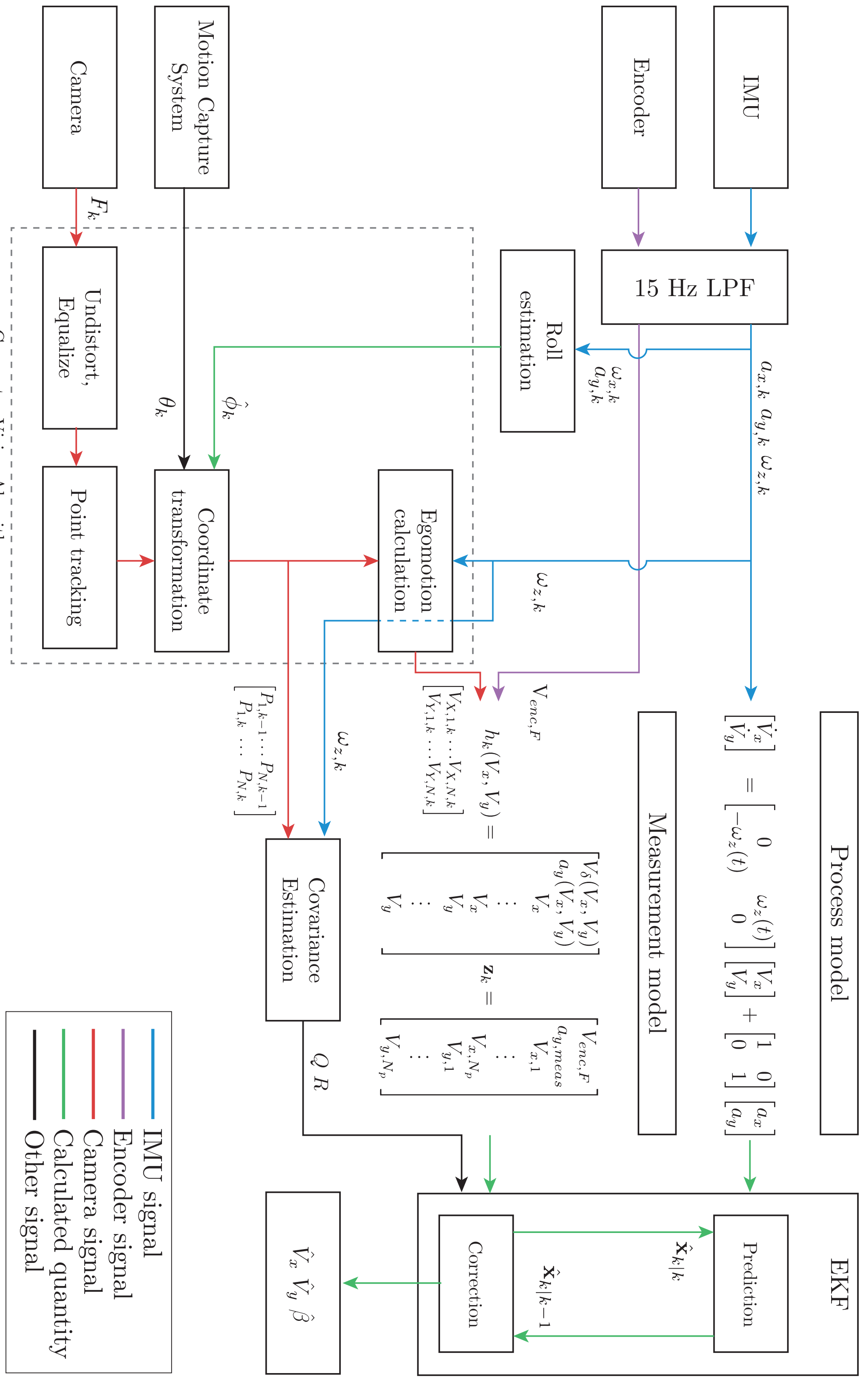

IMU
Encoder
15 Hz LPF
Roll estimation
Motion Capture System
Camera
$F_k$
Undistort, Equalize
Point tracking
Coordinate transformation
Egomotion calculation
Computer Vision Algorithm
$\theta_k$
$\hat{\phi}_k$
$\omega_{x,k}$ $a_{y,k}$
$a_{x,k}$ $a_{y,k}$ $\omega_{z,k}$
$\omega_{z,k}$
$V_{enc,F}$
Process model
$\begin{bmatrix}\dot{V}_x\\ \dot{V}_y\end{bmatrix} = \begin{bmatrix}0 & \omega_z(t)\\ -\omega_z(t) & 0\end{bmatrix}\begin{bmatrix}V_x\\ V_y\end{bmatrix} + \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}a_x\\ a_y\end{bmatrix}$
Measurement model
$h_k(V_x, V_y) = \begin{bmatrix}V_\delta(V_x,V_y)\\ a_y(V_x,V_y)\\ V_x\\ \vdots\\ V_x\\ V_y\\ \vdots\\ V_y\end{bmatrix}$ $\mathbf{z}_k = \begin{bmatrix}V_{enc,F}\\ a_{y,meas}\\ V_{x,1}\\ \vdots\\ V_{x,N_p}\\ V_{y,1}\\ \vdots\\ V_{y,N_p}\end{bmatrix}$
$\begin{bmatrix}V_{X,1,k}\dots V_{X,N,k}\\ V_{Y,1,k}\dots V_{Y,N,k}\end{bmatrix}$
$\begin{bmatrix}P_{1,k-1}\dots P_{N,k-1}\\ P_{1,k}\ \dots\ P_{N,k}\end{bmatrix}$
Covariance Estimation
$Q$ $R$
EKF
Prediction
Correction
$\hat{\mathbf{x}}_{k|k}$
$\hat{\mathbf{x}}_{k|k-1}$
$\hat{V}_x$ $\hat{V}_y$ $\hat{\beta}$
IMU signal
Encoder signal
Camera signal
Calculated quantity
Other signal

**Figure 5-3:** Full overview of the observer process, from the sensor input (left) up to the final estimates (bottom right)

## 5-3 Motion Capture System

To be able to identify parameters of the system, a set of validation data was needed. This data set was captured using a Motion Capture System (MCS) consisting of 10 Optitrack cameras, type Prime 13[61]. Within the arena, an area of 10 by 5 metres, the system can accurately track the position of a object with millimeter accuracy at 120 Hz. It does this by using a set of infrared lights, reflectors and cameras. Close to the ceiling of the arena infrared lights are mounted (on the camera modules) to send light towards the ground. The object to track then needs to be equipped with reflectors, which are also called retroreflective makers. An example of the BARC being equipped with these markers can be seen in Figure 5-4.

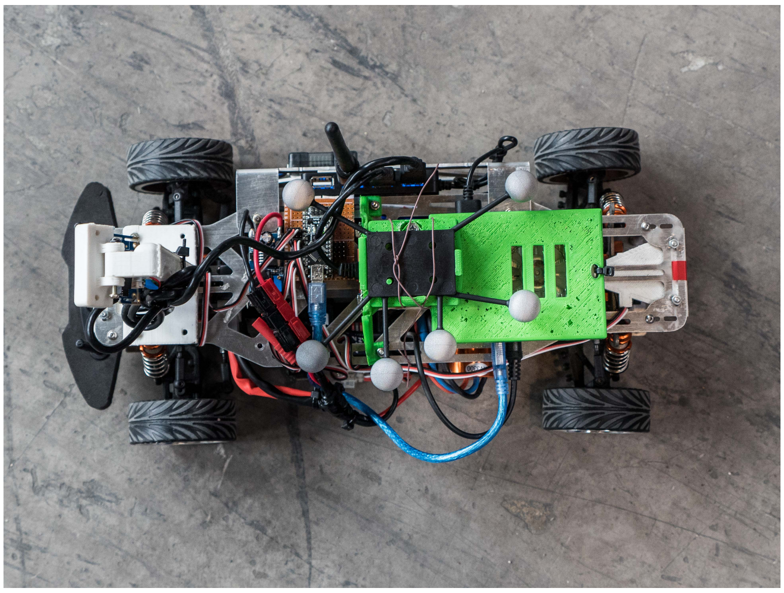


**Figure 5-4:** Picture of the BARC system seen from the top with the six retro reflective markers connected to a plastic structure.

The position and rotation of the center of the tracker, the piece of plastic holding all markers, is determined with an accuracy of around 1 mm in a global coordinate system. To obtain the vehicle's speed and acceleration in its local reference system, a coordinate transformation needs to be applied. The method for this was taken from [62]. The reference system of the global coordinates and the vehicle coordinates can be seen in Figure 5-5. To transform the position of an arbitrary point on the vehicle's body $q$ from body coordiantes $q_b$ to world coordinates $q_a$, transformation by the rotation matrix $R_{ab}$ and translation vector $p_{ab}$ are needed. This is done by:

$$q_a = R_{ab} q_b + p_{ab} \tag{5-1}$$

The rotation matrix $R_{ab}$ can be calculated based on the rotations measured by the MCS. It saves rotation in a Y-Z-X sequence and after correcting the signs of the axes and rotations to match a right hand system (see Figure 5-5), normal rotation matrices were used to calculate $R_{ab}$. This was done in the same way as for the camera homography from Section 3-1. The coordinate transformation can also be written in homogeneous form:

$$\begin{bmatrix} q_a \\ 1 \end{bmatrix} = \begin{bmatrix} R_{ab} & p_{ab} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} q_b \\ 1 \end{bmatrix} \tag{5-2}$$

$$\bar{q}_a = \bar{g}_{ab}\bar{q}_b \tag{5-3}$$

Where a bar above a symbol indicate a homogeneous form and $g_{ab}$ is the transformation matrix to transform coordinates from system $B$ to system $A$. Since a rotation matrix is orthonormal, its inverse is equal to the transpose of the matrix. The inverse of a transformation matrix $g$ can then be written out as:

$$g_{ab}^{-1} = g_{ba} = \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix} \tag{5-4}$$

Where $g_{ba}$ can be interpreted as the inverse of $g_{ab}$. That means that the first converts the coordinates of a point from system A to system B and the second does the operation in the other direction. The point that the MCS tracks is the center of the tracker holding the markers, which is not necessarily the center of gravity of the vehicle. Therefore the point $q$ from now on will be considered to be the center of gravity and the origin of $B$ the tracked center of the marker. To obtain the velocity $v_{q_b}$ of the point $q$ in reference system $B$ based on rotation and position measurements in system $A$, the following equations was used:

$$v_{q_b} = (R_{ab}^T \dot{R}_{ab}) \times q_b + R_{ab}^T \dot{p}_{ab} \tag{5-5}$$

Where $v_{q_b}$ is the resulting velocity of the centre of gravity and $q_b$ describes the vector from the center of the marker to centre of gravity. This was estimated to be $[0, 0, 0.04]$m ([X,Y,Z]), since the marker was put on a 3D printed mount that aligned the X- and Y-axis with the estimated location of the centre of gravity. Also a rotation offset of $[0, -0.8, 0.8]°$ ([roll,pitch,yaw]) was added to align the tracked marker rotations correctly with the vehicle's heading. This rotation offset was estimated by looking at the data of a maneuver where the BARC drove straight on and modifying the rotation offset until only a longitudinal velocity remained. Finally the acceleration in the x- and y-direction of the vehicle could be calculated by the following equation, assuming the vehicle exhibits only planar x-y motion:

$$a_x = \dot{V}_x + \omega_z V_y \tag{5-6}$$

$$a_y = \dot{V}_y - \omega_z V_x \tag{5-7}$$

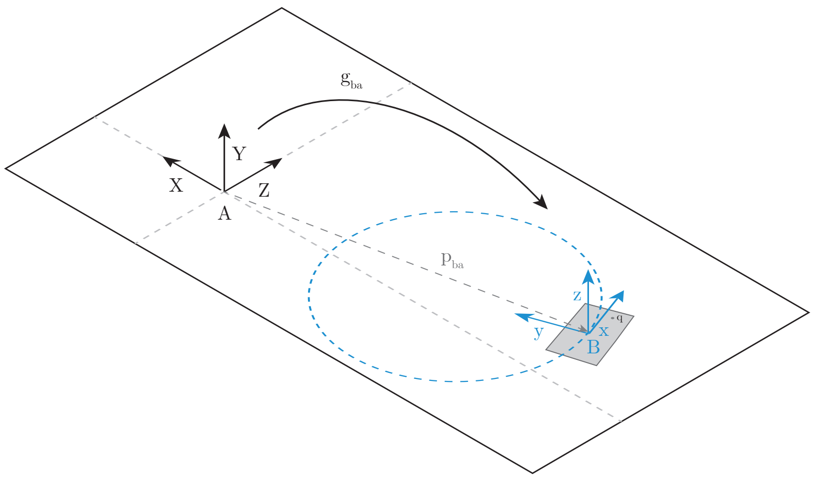


**Figure 5-5:** The arena with the global reference system A and the local vehicle reference system B. An arbirarty point p can be converted from system A to B with $g_{ab}$ and viceverse with $g_{ba} = g_{ab}^{-1}$, $p_{ab}$ is the translation vector between the origin of $A$ and $B$ in global coordinates.

## 5-4 System Identification

In this section the identification of some system parameters and relationships will be discussed that were required for implementing the proposed observer. First the parameters for the roll angle observer will be discussed. Then the relationship between the steering servo input and actual steering angle is presented. Finally the tire friction curves are identified, completing the system identification. Several dimensions and masses could be measured right away, which are listed in Appendix A-1.

### 5-4-1 Roll dynamics identification

As stated above, good estimation of the roll angle is vital to having useful camera data. In this section the parameters for the roll dynamics model and the observer covariance matrices will be estimated. The mass of the vehicle is known to be $m = 1.98\,\text{kg}$ and the sprung mass is $m_s = 1.7\,\text{kg}$. The moment of inertia around the $x$ axis is estimated by assuming the vehicle is a solid rod of 10 cm, the width of the center platform of the body. For a solid rod the moment of inertia is defined as:

$$I_{xx} = \frac{1}{12} m L^2 = 1.4 \times 10^{-3}\,\text{N}\,\text{m}^2 \tag{5-8}$$

The remaining parameters to be estimated are $h_s, K_\phi, b_\phi, Q, R$. Process noise covariance matrix $Q$ is defined as a diagonal matrix with elements $q_1, q_2$, measurement noise covariance matrix $R$ is just a single value $r_1$, since only one signal is used as a measurement. To fit all six parameters, a nonlinear optimization problem was solved. Optimizing the six parameters to lower the Normalized Root Mean Square Error (NRMSE) between validation data and observer output using the grey-box identification function from Matlab. The NRMSE is similar to the Variance Accounted For (VAF) and defined as:

$$\text{NRMSE} = \left(1 - \sum_{i=1}^{N} \frac{||\mathbf{x} - \hat{\mathbf{x}}||_2}{||\mathbf{x} - \bar{\mathbf{x}}||_2}\right) \cdot 100\% \tag{5-9}$$

Where $\mathbf{x}$ represents the validation data, $\hat{\mathbf{x}}$ the observer output, $\bar{\mathbf{x}}$ the mean of the validation data and $||\ldots||_2$ the 2-norm. The grey box optimization problem is then formulated as:

$$\begin{aligned} \underset{h_s, K_\phi, b_\phi, q_1, q_2, r_1}{\text{minimize}} \quad & 1 - \sum_{i=1}^{N} \frac{||\mathbf{x} - \hat{\mathbf{x}}_{k|k}||_2}{||\mathbf{x} - \bar{\mathbf{x}}||_2} \\ \text{subject to} \quad & \hat{\mathbf{x}}_{k+1} = (A - KC)\hat{\mathbf{x}}_k + B\mathbf{u}_k + K_k \mathbf{z}_k \quad i = 1, \ldots, N. \end{aligned}$$

The identification was done on a maneuver consisting of a 0.5 Hz sinusoidal steering input with amplitude 13° and a constant rear wheel torque, maneuver 1A. The input of the optimization were the onboard measured lateral acceleration $a_y$ and the measured roll velocity $\omega_x$, put through a 5 Hz low-pass filter. Measurements coming from the MCS, see Section 5-3, were used as validation data. Results can be seen in Figure 5-6 and Table 5-1.

Validation was then performed on two different maneuvers. Another sinusoidal steering input, but of twice the frequency and a ramp steering input, starting at 7.5° and increasing with

**Table 5-1:** Estimated parameters for roll dynamics observer

| Parameter | Value | Unit |
|---:|---:|---:|
| $h_s$ | 0.084 | m |
| $K_\phi$ | 15.2378 | $\mathrm{N\,m\,rad^{-1}}$ |
| $C_\phi$ | 0.8410 | $\mathrm{N\,m\,s\,rad^{-1}}$ |
| $q_1$ | 0.10 | - |
| $q_2$ | 46.00 | - |
| $r_1$ | 100.0 | - |

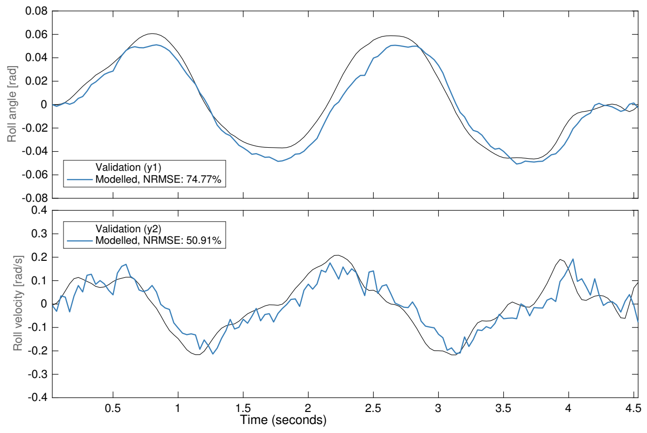
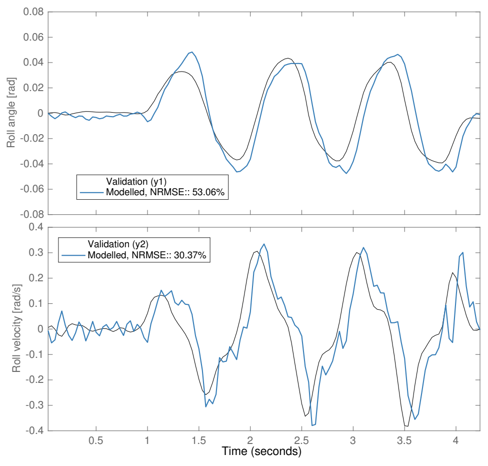
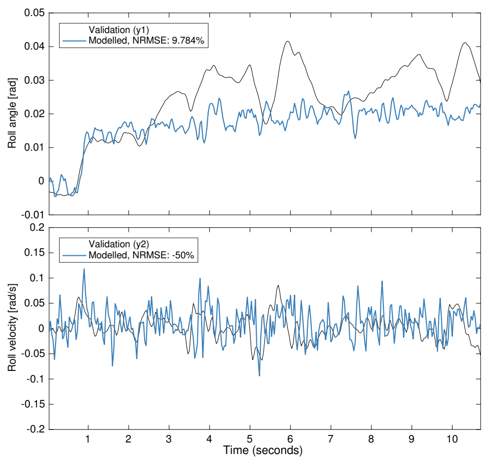


**Figure 5-6:** Estimation of roll dynamics on a maneuver a sinusoidal steering input of $1\,\mathrm{Hz}$ and constant rear wheel torque.

0.3° per second. Results of this can be seen in Figure 5-7. When the roll angle is changing a lot due to steering input, roll velocity is high and the observer is able to follow the measured roll angle well. When the vehicle is driving in near steady state condition as in Figure 5-7b, roll velocity decrease and it becomes harder for the observer to track the roll angle. However, as will later be presented in the next chapter, the current observer gives results that are good enough for correction of the homography matrix.



**(a)** Validation of roll dynamics on a maneuver applying a sinusoidal steering input of 1 Hz and constant rear wheel torque.



**(b)** Validation of roll dynamics on a maneuver using a ramping steering input of $10°$-$15°$ and constant rear wheel torque.

**Figure 5-7:** Validation of the estimate of roll angle and roll velocity coming from the Kalman filter on two maneuvers different to the one used for estimation of parameters.

## 5-4-2 Steering angle

Steering of the of the BARC is done by a servo motor connecting to the steering rod of the vehicle. The servo motor is controlled by a Pulse Width Modulation (PWM) signal ranging from 0 to 180, where 0 indicates the servo steering fully to the right, 90 indicating neutral steering and 180 indicating fully steering to the left. During operation of the algorithm, the front axle steering angle $\delta_f$ is needed for the calculation of $V_\delta(V_x, V_y)$ and the wheel slip angles $\alpha_f, \alpha_r$. To estimate $\delta_f$ during operation a mapping was made from the PWM signal to a steering angle.

This mapping was made by having the vehicle drive in near steady state conditions and calculating the steering angle. The near steady state conditions were achieved by applying a medium constant rear wheel torque and slowly increasing the steering angle between different ranges. The steering angle $\delta_f$ was calculated by assuming an Ackermann steering geometry, which implies that the lines perpendicular to the direction of the wheels intersect at the radius of a turn. Therefore $\delta_f$ can be approximated as:

$$\delta_f \approx \arctan\left(\frac{\omega_z(l_f + l_r)}{V_x}\right) \tag{5-10}$$

This method was taken from [63]. The results for 6 experiments, three left hand and three right hand turns, can be seen in Figure 5-8. The function fitted to the relationship of the PWM input of the steering servo and the calculated steering angle is:

$$\hat{\delta}_f = \min\Big(\max(9.55 \cdot 10^{-3}(x - 90) + 3.67 \cdot 10^{-3}, -0.24), 0.30\Big) \tag{5-11}$$

Where the steering angle $\delta_f$ is given in radians. This was fitted using a nonlinear optimization algorithm by letting it optimize the steepness $a$ and the vertical shift $b$ of the affine function $f(x) = ax + b$. The saturation points roughly were estimated to a $\pm 5°$ margin and then also optimized. On a set of 5911 data points the RMS value of the error was 0.0177 radians.

Next to modelling the linear relationship from servo input to steering, the time delay between sending a PWM command and seeing a change in steering angle must be considered too. The used servo has a known rotational speed of 60° per second. To measure the delay a square wave of $\delta = \pm 14°$ and $f = 1\,\text{Hz}$ is used as a steering input, and in similar fashion to the steering law identification, the steering angle $\delta_f$ was calculated using (5-10). In Figure 5-9 it can be observed that a delay of three samples gives an adequate compensation for the servo delay, when sampled at 30 Hz. The calculated steering angle also seems to have some transient dynamics going on, but this is probably because of (5-10) assuming steady state behavior, which is not the case when the steering angle just changed 26°.

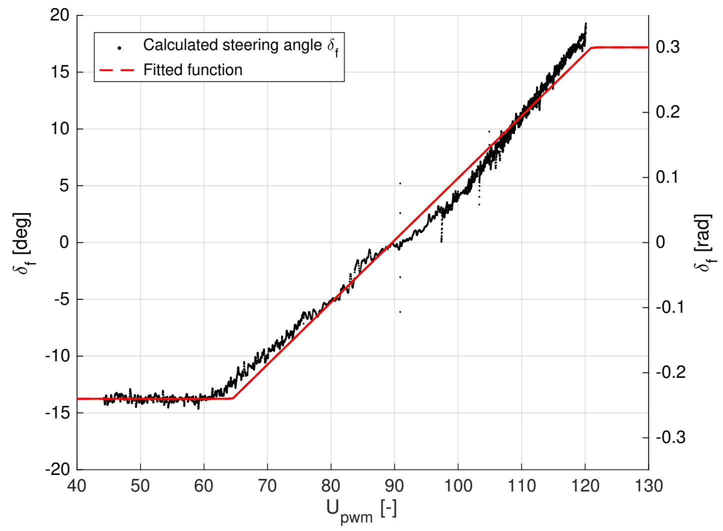


**Figure 5-8:** Calculated steering angle for different servo commands and the fitted function for converting servo commands to steering angle. All data points were gathered on ramp steer maneuvers with $\Delta\delta_f/\Delta t < 0.5° s^{-1}$. Number of points $= 5911$

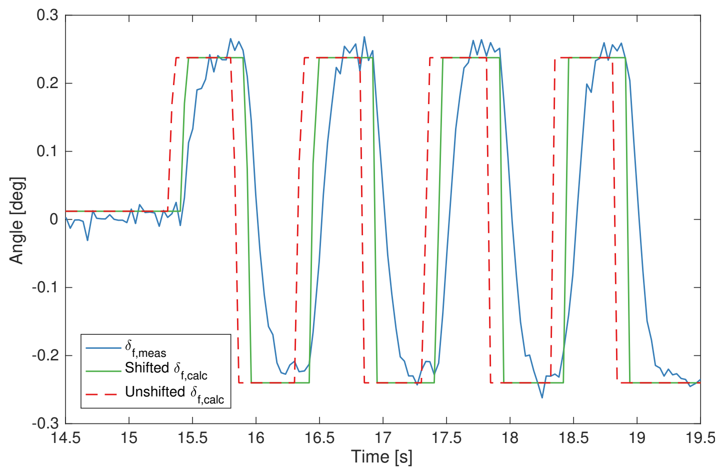


**Figure 5-9:** Comparison of measured steering angle, the shifted calculated steering angle and the unshifted calculated steering angle. The calculated steering angle is shifted by three samples at a sample rate of $30\,\text{Hz}$. Performed maneuver is a $1\,\text{Hz}$ square wave steering input of $\delta_f = \pm 14°$ and a constant torque.

## 5-4-3 Tire model

The final step in system identification is identifying the relationship between wheel slip angle and lateral tire force, so that it can be used for calculated the lateral acceleration based on the vehicle's longitudinal and lateral velocity using (2-12). The model to be fitted is the hyperbolic tangent model:

$$F_{yi} = \frac{C_i}{k_i} \tanh\left(k_i \alpha_i\right) \quad i = \{f, r\} \tag{5-12}$$

$$\alpha_f = \left(\delta_f - \arctan\left(\frac{V_y + l_f \omega_z}{V_x}\right)\right) \ \alpha_r = -\arctan\left(\frac{V_y - l_r \omega_z}{V_x}\right) \tag{5-13}$$

The model assummes no combined slip and lumps the left and right wheels together for each respective axle. To identify the tire forces the vehicle performed several quasi steady-state maneuvers where the yaw acceleration is low. Therefore it is assumed that the lateral forces are balanced:

$$0 = l_f F_{yf} - l_r F_{yr} \tag{5-14}$$

By measuring the lateral acceleration, both tire forces can be calculated:

$$F_{yf} = \frac{m a_y l_r}{l_f + l_r} \quad F_{yr} = \frac{F_{yf} l_f}{l_r} \tag{5-15}$$

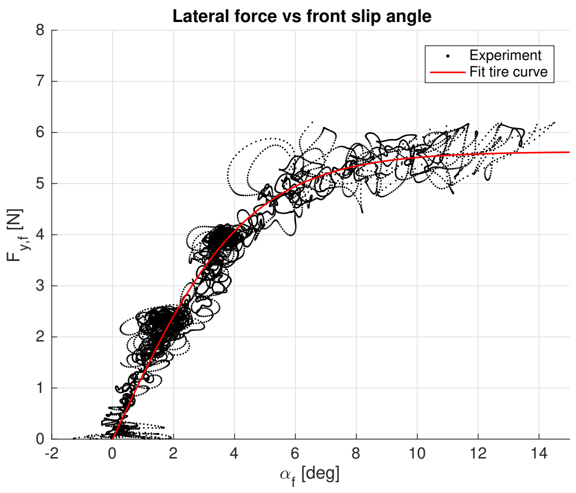
This method was taken from [11]. For all experiments the lateral acceleration, both velocities and the yaw rate are measured at $120\,\text{Hz}$ using a Motion Capture System. For details about the MCS see Section 5-3. For all used maneuvers, the tire forces and the wheel slip angles are calculated and plotted against each other. The used maneuvers are:

- $2\times$ fixed steering angle of $10°$, ramp up rear wheel torque from low to high setting (0-4N), total duration 18s.
- $1\times$ fixed medium-low rear wheel torque (1.75N), ramp up steering angle from $7°$ to $17°$ at $0.33°$ per second, total duration 30 seconds.
- $1\times$ fixed medium rear wheel torque (2.25N), ramp up steering angle from $7°$ to $17°$ at $0.33°$ per second, total duration 30 seconds.

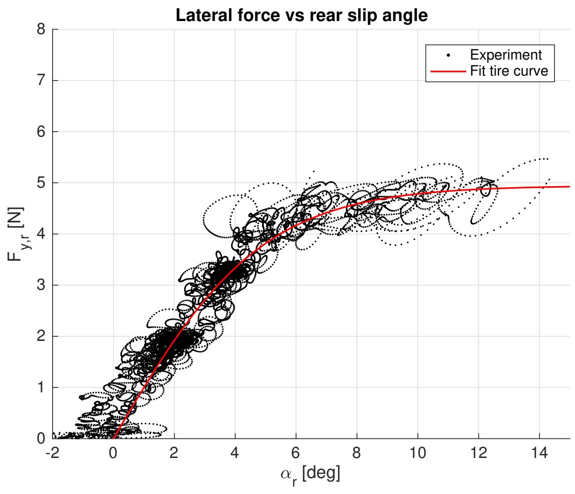
All maneuvers were left hand turns. Although it would be better to identify turns in both directions, it was not considered a real problem if validated later against turns in both directions. The results of the identification can be seen in Figure 5-10 and the parameters are found in Table 5-2. The hyperbolic tangent model was fit by solving a nonlinear optimization problem using a Trust-Newton algorithm. The Root Mean Square Error (RMSE) between the fit and the original data points ($n = 14428$) was 0.4866 N for $F_{yf}$ and 0.3859 N for $F_{yr}$.

**Table 5-2:** Tire model parameters found for front and rear axle

| | $\mathbf{C_i}$ | $\mathbf{k_i}$ |
|---|---|---|
| Front | 74.03 | 12.66 |
| Rear | 57.76 | 11.72 |



**(a)** Front wheel tire force vs slip angle. Fitted tire model in red, $C_f = 74.03$ and $k_f = 12.66$,



**(b)** Rear wheel tire force vs slip angle. Fitted tire model in red, $C_f = 57.76$ and $k_f = 11.72$, RMS error of 0.3859 to the original data points

**Figure 5-10:** Calculated total lateral forces for the front axle (left) and rear axle (right) versus wheel slip angles. On top of the data points is the fit hyperbolic tangent model. Number of points $= 14428$.

After the identification, validation was performed to ensure that the fitted function can be used to calculate the lateral acceleration and to validate right and left turn behavior. For calculating the acceleration the longitudinal and lateral speed from the validation data set was used together with the onboard recorded lateral acceleration and yaw rate. Results can be seen in Figure 5-11.

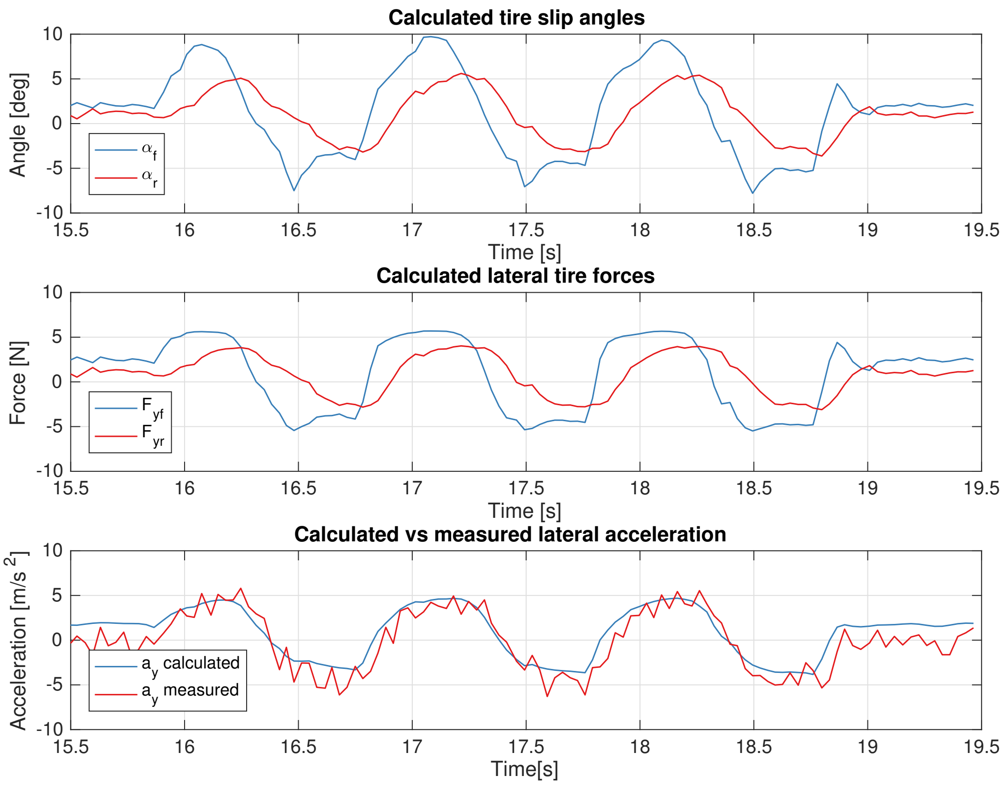


**Figure 5-11:** Comparison of measured lateral acceleration and the calculated lateral acceleration (bottom). The calculated lateral acceleration is based on calculated slip angles (top) and calculated forces (middle). Performed maneuver is a sinusoidal steering input of $1\,\mathrm{Hz}$ with an amplitude of $\delta_f = 20°$ combined with a constant rear wheel torque (maneuver 3C).

## 5-5 Conclusion

The final steps to implement the observation methodology have been discussed in this chapter. A low-pass filter was added to prevent aliasing when downsampling the IMU and encoder signals. The relevant system parameters were estimated using the data captured by the Motion Capture System (MCS). The roll dynamics, steering law and tire force parameters are know known and the observer can be evaluated using experiments.

# Chapter 6

# Experiments

In this chapter the performance of the observation methodology will be evaluated using experiments. First the experiment setup will be discussed by presenting the type of open loop maneuvers performed and the testing location. Then the tuning of the Dynamic Covariance Estimation (DCE) and Extended Kalman Filter (EKF) will be evaluated using a sensitivity analysis on select maneuvers. Once those two sections are done, the results for all different maneuvers are presented. The chapter is concluded by looking at the robustness of the observer by varying some model parameters and looking at its effect on estimation performance.

## 6-1 Experiment setup

In the previous chapter the implementation of the observer was established. Open loop maneuvers are performed on the Berkeley Autonomous Race Car (BARC), after which the data will be taken to a laptop and the observer will be run. Different types of maneuvers were preprogrammed. Three maneuvers were selected to be used in this report:

- A sine steer maneuver: the rear wheel torque is kept constant and after a delay a sinusoidal signal is put on the steering servo. In this way a maneuver with a high excitation on the lateral velocity and lateral acceleration is created. This is a baseline maneuver on which the kinematic model based observer should perform well.
- A ramp steer maneuver: the rear wheel torque is again kept constant, but now the steering angle is slowly increased after a delay. A quasi steady state maneuver is created. These types of maneuvers are harder for the kinematic model based observer. Here the camera and calculated lateral acceleration should come into play.
- A sine with dwell maneuver with reduced rear wheel grip: the sine with dwell maneuver is a standardized maneuver to test vehicle lateral stability. In the experiments using this maneuver, plastic tape is applied to the rear wheels to reduce rear cornering stiffness. The vehicle starts again with a constant rear wheel torque. After a delay a sinusoidal

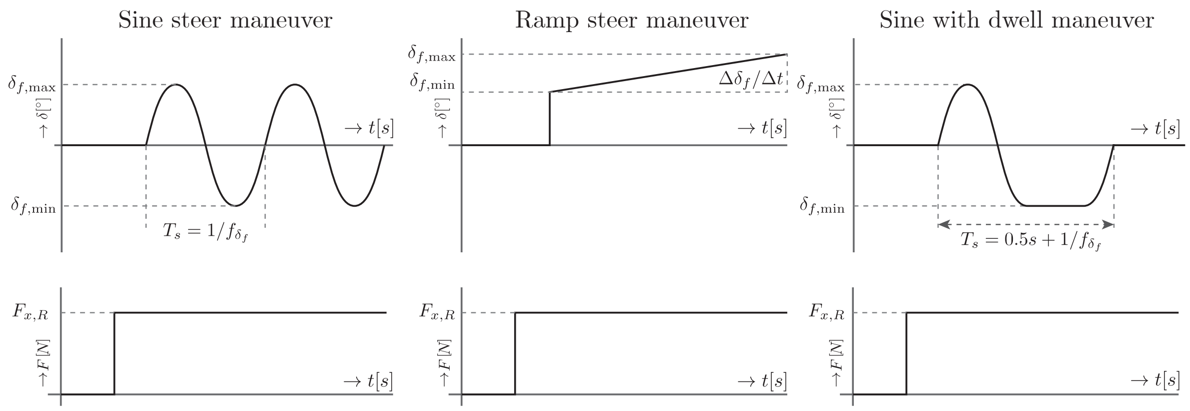


**Figure 6-1:** Overview of the three types of maneuvers used in this chapter and the parameters that can be edited of the maneuvers

steering input of 0.7 Hz is applied. At the second maximum steering input (so a left-right or right-left maneuver) the maximum steering input is held for 0.5 s. This should allow the vehicle to spin and enter the nonlinear driving regime.

A graphical representation of the three used maneuvers can be found in Figure 6-1. Throughout the whole report these maneuvers will be referenced towards as a number-letter combination. The number indicates which type of maneuver and parameter set is used and the letter indicates which repetition of the maneuver is referenced. These combinations and their open loop parameters can be found in Table 6-1.

The experiments were performed in the testing arena where the Motion Capture System (MCS) was also mounted. The floor of this arena was generally flat, but some bumps were present. Also the floor offered a modest contrast. Some white lines and white sheets of paper were present on the ground. These were used for other experiments and could not be removed. The amount and type of markers on the ground were considered to have enough similarities with the line markings of an actual road. A picture of the arena can be found in Figure 6-2.

**Table 6-1:** Six different types of open loop maneuvers used in this report, including the input parameters.

| **Ref.** | **Type** | $F_{x,R}$ [N] | $f_{\delta_f}$ [Hz] | $\Delta\delta_f/\Delta t$ [°/$s$] | [$\delta_{f,\min}, \delta_{f,\max}$] [°] |
|---|---|---|---|---|---|
| 1A - 1C | Sine steer | 1.75 | 0.50 | - | [-13, 13] |
| 2A - 2C | Sine steer | 2.20 | 0.50 | - | [-13, 13] |
| 3A - 3C | Sine steer | 2.20 | 1.00 | - | [-13, 13] |
| 4A - 4C | Ramp steer | 2.20 | - | 0.33 | [7.5, 16.5] |
| 5A - 5C | Ramp steer | 2.20 | - | 0.33 | [7.5, 16.5] |
| 6A - 6C | Sine with dwell | 2.75 | 0.70 | - | [-13, 13] |

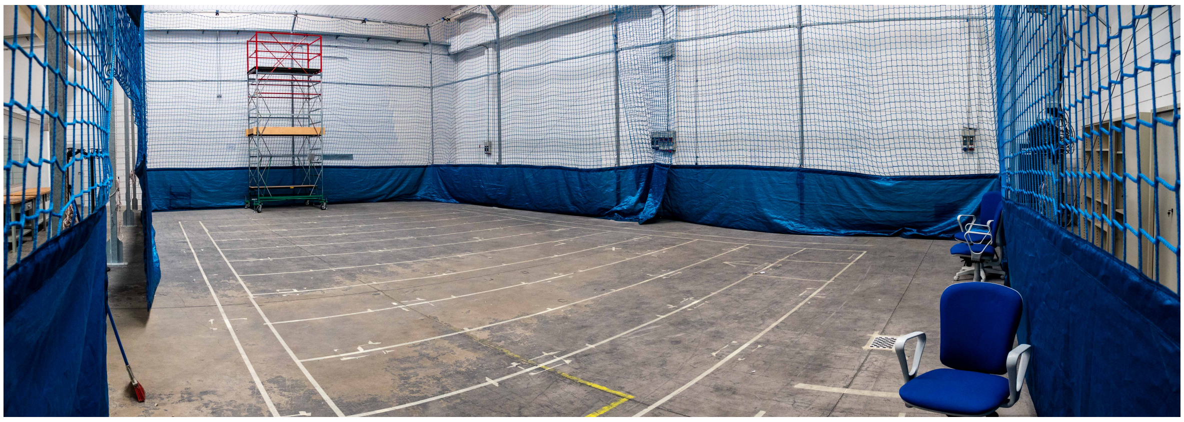


**Figure 6-2:** Panorama of the testing arena and an indication of the ground surface used. Note: on the day of the actual tests around 30 white sheets of A4 paper were also present on the ground plane in a grid pattern spread over the whole arena.

## 6-2 Filter tuning

Before all results can be presented, the tuning of the DCE and EKF is discussed first. The hybrid dynamic-kinematic model based observer will be tested in three variations, in order to assess the performance of different subparts separately:

1. With Computer Vision Algorithm (CVA) and DCE, so the full observer with camera, Inertial Measurement Unit (IMU) and wheel speed sensor to assess the full potential performance of the developed methodology.
2. With CVA but without DCE, to assess the effect DCE has on performance. This will be the exact same observer as variation one, except for the camera measurement noise covariance submatrix which is now static.
3. Without CVA and without DCE, to establish a baseline performance of a good state-of-the-art observer that only uses the IMU and wheel speed sensor.

In this section the tuning of the first observer will be discussed in depth with some sensitivity analysis, for the other two the tuning will just be given, since the process works in the same way and the same trade-offs need to be made.

Tuning a filter is a process that is usually based on heuristics. To give more insight in the process used for this project, the following approach is taken in this section: First the available sensor signals are shown for one run, to get an idea of the data that is used for estimation. Then the sensitivity of some parameters of the DCE is shown, followed by the sensitivity of the other measurement noise covariance matrix parameters and finally completed by the sensitivity of the process noise covariance matrix parameters. As stated before, these covariance matrices $Q$ and $R$ are assumed to be diagonal and contain the following parameters:

$$Q = \begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix} \qquad R_k = \begin{bmatrix} r_{enc} & 0 & 0 & 0 \\ 0 & R_{V_{x,k}} & 0 & 0 \\ 0 & 0 & R_{V_{y,k}} & 0 \\ 0 & 0 & 0 & r_{a_y} \end{bmatrix} \tag{6-1}$$

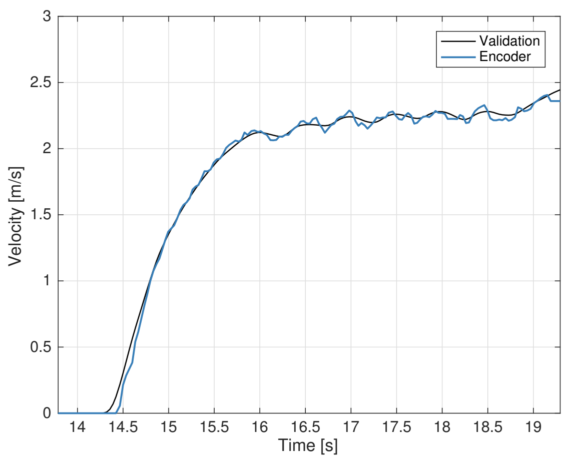
Where $R_{V_x}, R_{V_y}$ are diagonal submatrices that contain the results of the DCE with dimensions equal to $\mathbb{R}^{N_p \times N_p}$, where $N_p$ equals the number of points found in that frame. This results in an $R_k$ of size $\mathbb{R}^{2+2N_p \times 2+2N_p}$. In the case of the observer with CVA but without DCE submatrices $R_{V_x}, R_{V_y}$ will consist of constant diagonal entries $r_{V_x}, r_{V_y}$ and still have the same dimensions as the matrices for the observer with DCE.

A summary of the DCE can be found in Section 4-5. There, it was established that only $\sigma^2_{\text{hom}}, \sigma^2_{\text{track}}, \alpha$ and $\sigma^2_{\omega,z}$ have to be tuned. These are respectively the base variance for homography error scaling, base variance for tracking error scaling, the parameter defining the affine sum of the two tracking error scaling functions and the variance of the yaw rate measurement noise.

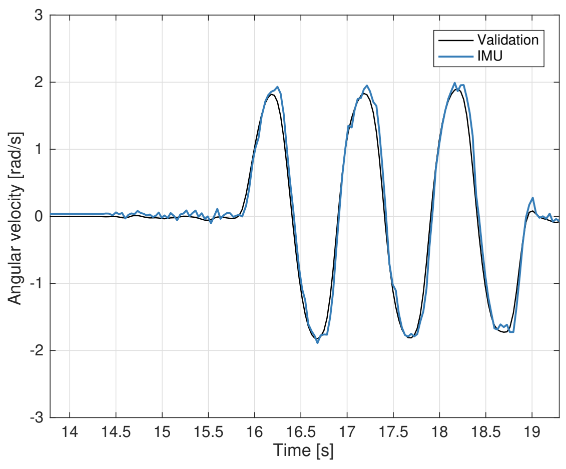
For experiment 3C the available sensor signals can be seen in Figure 6-3. It can be seen that the encoder velocity measurement follows the validation speed quite accurately. The same can be said for the yaw rate measurement. The latency of both measurements is small and accuracy is good. The acceleration measurements however are quite noisy, as is generally the case for acceleration sensors. Finally the camera measurements tracked the validation velocity acceptably, however for the longitudinal velocity there is some offset. This can be attributed to the fact that the pitch angle is not estimated online. The vehicle tilts down roughly $1^\circ$ during cornering, which will result in pixels being located higher up in the frame than they should be, thus increasing their longitudinal velocity. The lateral velocity roughly follows the validation speed, yet there is a lot of noise present on the signal. These six sensor signal plots show that the sensors track the validation quantities to a certain degree, but an EKF is needed to produce an accurate, noise free estimate. This is where the EKF comes into play. After testing a lot of variations, the overall tuning of the DCE and EKF ended up as follows:

**Table 6-2:** Tuning of the three different observers compared in the results

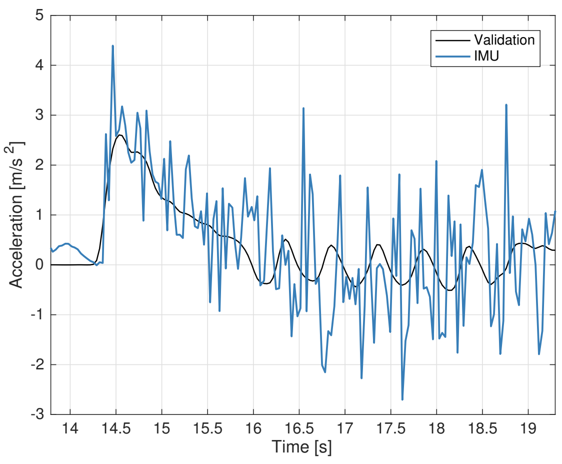
| | Observer type | | |
|---|---|---|---|
| Variable | **Camera with DCE** | **Camera without DCE** | **Without camera** |
| $q_1$ | 0.05 | 0.05 | 0.01 |
| $q_2$ | 0.025 | 0.025 | 0.01 |
| $r_{enc}$ | 0.5 | 0.5 | 0.1 |
| $r_{a_y}$ | 150 | 150 | 5 |
| $r_{V_x}$ | - | 20 | - |
| $r_{V_y}$ | - | 20 | - |
| $\sigma^2_{\text{hom}}$ | 2500 | - | - |
| $\sigma^2_{\text{track}}$ | 0.050 | - | - |
| $\alpha$ | 0.7 | - | - |
| $\sigma^2_{\omega,z}$ | 2 | - | - |



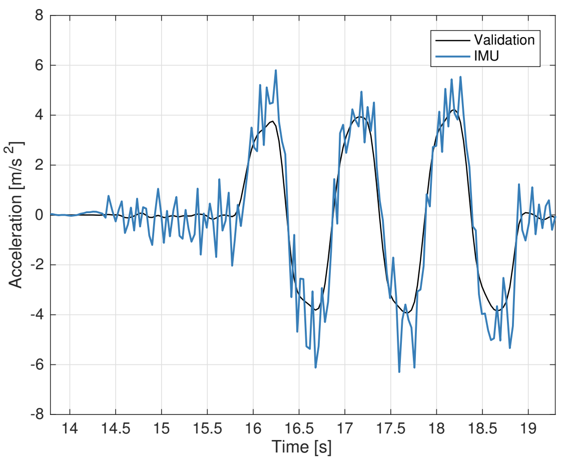
**(a)** Encoder velocity



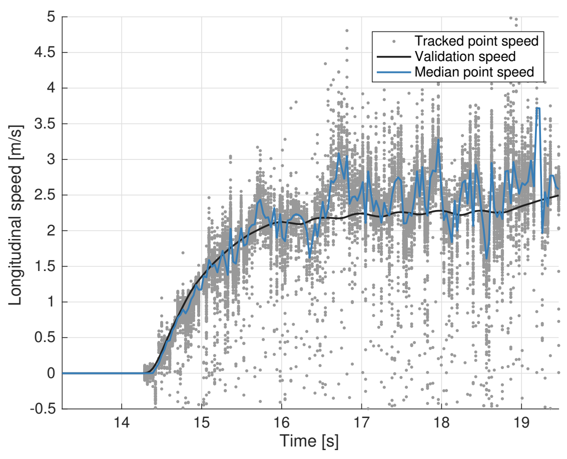
**(b)** IMU yaw rate



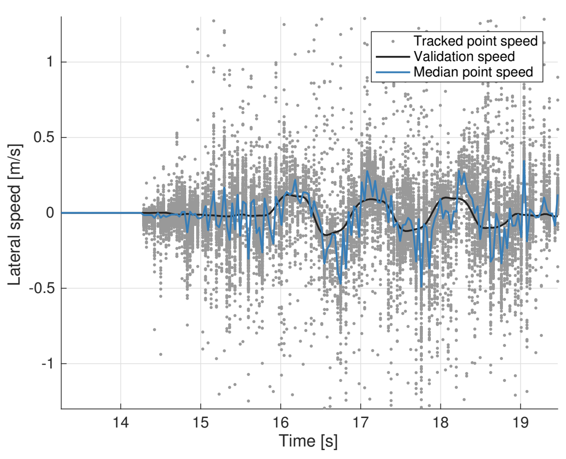
**(c)** IMU longitudinal acceleration



**(d)** IMU lateral acceleration



**(e)** Camera longitudinal velocity

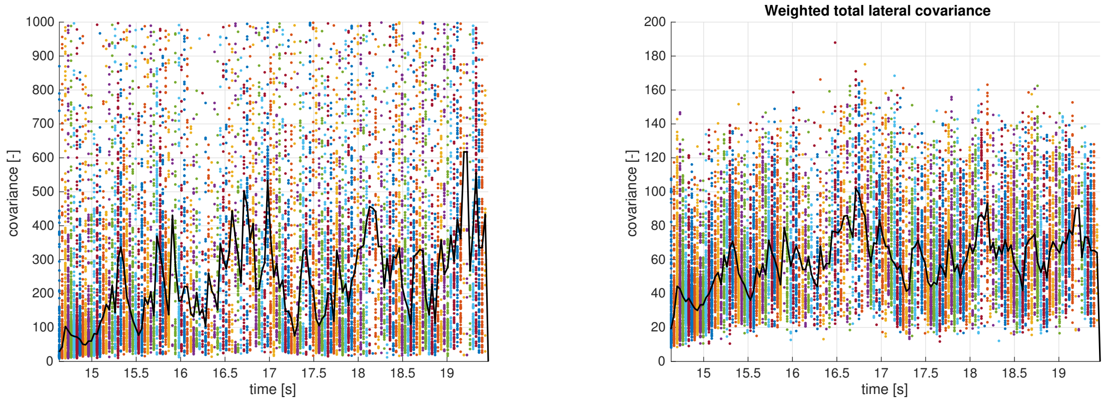


**(f)** Camera lateral velocity

**Figure 6-3:** Available sensor signals for sine steer run 3C, $f = 1.0\,\mathrm{Hz}, F_{xR} = 2.2\mathrm{N}$

## 6-2-1 Dynamic Covariance Estimation tuning

The first step in tuning the filter was setting the parameters for the DCE. The DCE was mostly needed for the lateral speed estimation, since the longitudinal speed estimation could also rely on the front wheel speed sensor in most cases. Therefore the parameters $\sigma^2_{\text{hom}}, \sigma^2_{\text{track}}, \alpha$ and $\sigma^2_{\omega,z}$ were tuned to give good performance on the lateral speed estimation. The noise variance for $V_{x,i}$ was seen as a result of that and not altered any further. Good results were obtained when the contribution from $\bar{\sigma}^2_{i,y,k-1}$ and $\bar{\sigma}^2_{i,y,k}$ in the variance of one lateral velocity $V_{i,y,k}$ had a ratio of roughly 1:7. To see the results of these parameters, a scatter plot of all calculated scaled noise variances for maneuver 3C can be seen in Figure 6-4.



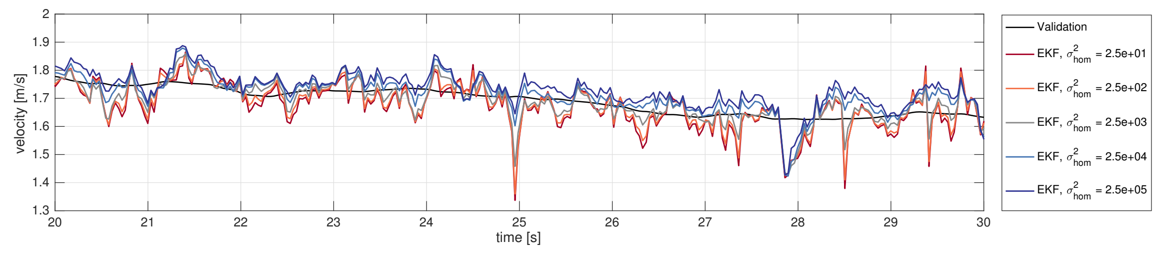
**(a)** Calculated longitudinal velocity variance **(b)** Calculated lateral velocity variance

**Figure 6-4:** Calculated longitudinal and lateral camera velocity variance for all tracked points. Each frame has a new color and the median per frame is plotted in black

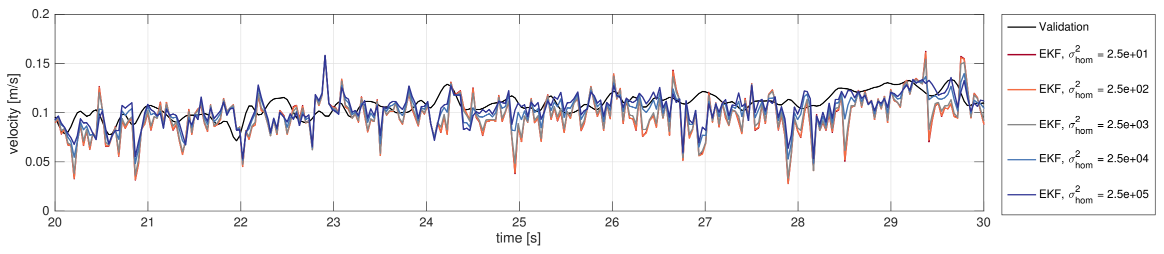
To get a deeper insight in how changing the parameters of the DCE influences the final estimate, a sensitivity analysis was done for base variances $\sigma^2_{\text{hom}}, \sigma^2_{\text{track}}$ on maneuver 4A. Maneuver 4A is a quasi steady-state maneuver, where the kinematic part of the observer has difficulties following the true velocities and the camera measurements should aid estimation.

The results for the sensitivity of $\sigma^2_{\text{hom}}$ can be seen in Figure 6-5 and the performance metrics are located in Table 6-3. Generally it can be seen that a lower base variance $\sigma^2_{\text{hom}}$ results in the observer following the camera signal more closely, which is noisier and sometimes has an offset. On the longitudinal velocity it can be seen that if $\sigma^2_{\text{hom}}$ is on its medium setting, $2.5 \cdot 10^3$, it delivers the best estimate for longitudinal velocity. This is probably due to the encoder velocity having a small positive offset. Having the observer balance between the camera and encoder signal gives the best results.

For the lateral velocity a higher $\sigma^2_{\text{hom}}$ makes the observer rely more on the calculated lateral acceleration correction, which is very accurate for these kind of quasi steady-state maneuvers. Having the observer trust the calculated lateral acceleration more comes at the cost of less robustness when the lateral acceleration calculation is not correct. This can happen due to errors in the used vehicle and tire model. These errors might come from: a poorly estimated vehicle mass, not up to date tire friction estimation or a too large yaw acceleration. The trade-off between steady-steady performance and robustness to uncertainty in the vehicle model is



**(a)** Longitudinal velocity sensitivity for $\sigma^2_{\text{hom}}$



**(b)** Lateral velocity sensitivity for $\sigma^2_{\text{hom}}$

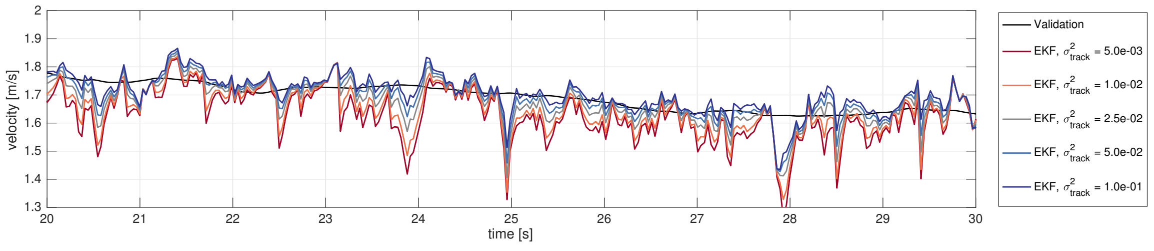
**Figure 6-5:** Longitudinal and lateral velocity estimate sensitivity for $\sigma^2_{\text{hom}}$ tuning in Dynamic Covariance Estimation (DCE) on run 4A

demonstrated when setting the measurement noise variance of the lateral acceleration signal in Subsection 6-2-3.

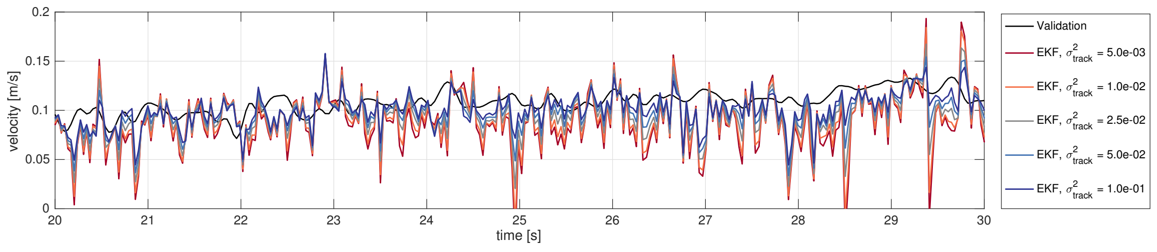
The same observations can be made when assessing the sensitivity of $\sigma^2_{\text{track}}$. Results of this analysis are shown in Figure 6-6 and Table 6-3. A higher base variance results in better observer performance on the longitudinal and lateral velocity estimates. As $\sigma^2_{\text{track}}$ increases, the estimates gets closer to the black validation velocity line. This is the same effect that was observed when tuning $\sigma^2_{\text{hom}}$. No other downsides to increasing the velocity can be observed, except for the same loss of robustness, which will be discussed later. It was found that a combination of $\sigma^2_{\text{hom}} = 2500, \sigma^2_{\text{track}} = 0.050$ was a good starting point for tuning the other parameters of the EKF.

**Table 6-3:** Performance metrics for sensitivity analysis of DCE tuning on run 4A

| | Estimated $V_x$ | | Estimated $V_y$ | | | Estimated $V_x$ | | Estimated $V_y$ | |
|---|---|---|---|---|---|---|---|---|---|
| $\sigma^2_{\text{hom}}$ | RMSE | VAF | RMSE | VAF | $\sigma^2_{\text{track}}$ | RMSE | VAF | RMSE | VAF |
| $2.5 \cdot 10^1$ | 0.0686 | 88.29% | 0.0252 | 79.31% | $5.0 \cdot 10^{-3}$ | 0.1019 | 83.55% | 0.0383 | 55.69% |
| $2.5 \cdot 10^2$ | 0.0626 | 90.00% | 0.0251 | 79.46% | $1.0 \cdot 10^{-2}$ | 0.0838 | 87.27% | 0.0344 | 63.77% |
| $2.5 \cdot 10^3$ | 0.0535 | 92.31% | 0.0235 | 81.83% | $2.5 \cdot 10^{-2}$ | 0.0627 | 90.86% | 0.0281 | 75.08% |
| $2.5 \cdot 10^4$ | 0.0547 | 92.79% | 0.0189 | 87.14% | $5.0 \cdot 10^{-2}$ | 0.0535 | 92.31% | 0.0235 | 81.83% |
| $2.5 \cdot 10^5$ | 0.0660 | 92.39% | 0.0172 | 88.37% | $1.0 \cdot 10^{-1}$ | 0.0514 | 92.92% | 0.0200 | 85.90% |



**(a)** Longitudinal velocity sensitivity for $\sigma^2_{\text{track}}$

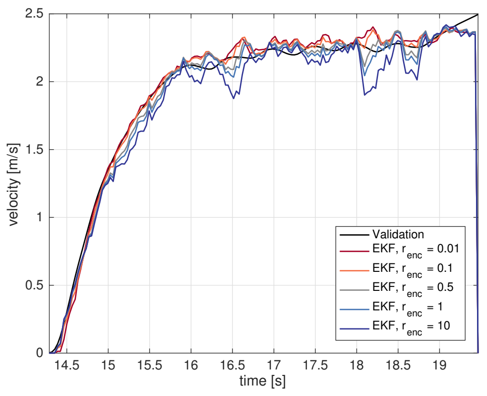


**(b)** Lateral velocity sensitivity for $\sigma^2_{\text{track}}$

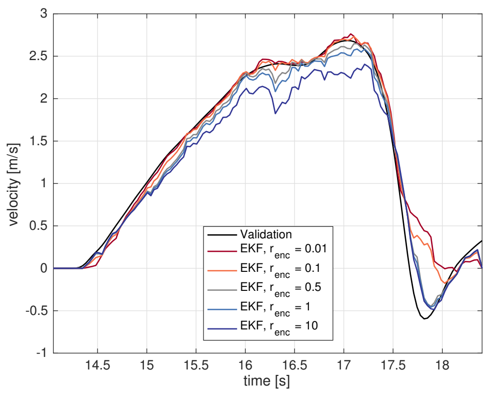
**Figure 6-6:** Longitudinal and lateral velocity estimate sensitivity for $\sigma^2_{\text{track}}$ tuning in Dynamic Covariance Estimation (DCE) on run 4A

### 6-2-2 Encoder measurement variance

The second noise variance to be tuned was the encoder speed noise variance. The encoder can provide reliable, relatively noise free results as long as wheel slip is small. To illustrate this, five variations of $r_{enc}$ have been simulated on maneuver 3C (a linear driving maneuver) and 6A (a nonlinear driving maneuver). Results of this can be seen in Figure 6-7 and Table 6-4. On normal maneuvers, such as maneuver 3C, decreasing the variance will in general result in better estimates of the longitudinal velocity. This is because there will be more weight put on the correction made based on the wheel encoder measurement. However, setting $r_{enc}$ too low will affect estimation performance when the vehicle is slipping, for example in maneuver 6A. The vehicle makes a 180° spin and drives backwards for a small amount of time. In Figure 6-7b it can be seen that once $r_{enc}$ gets below 0.50 the encoder is trusted too much and estimates become inaccurate. The encoder cannot distinguish between forward and backward motion, so the velocity measurement is incorrect. Tuning $r_{enc}$ to 0.50 struck a good balance between accuracy in normal maneuvers and robustness when drifting.



**(a)** Longitudinal velocity sensitivity for $r_{enc}$ on run 3A



**(b)** Longitudinal velocity sensitivity for $r_{enc}$ on run 6A

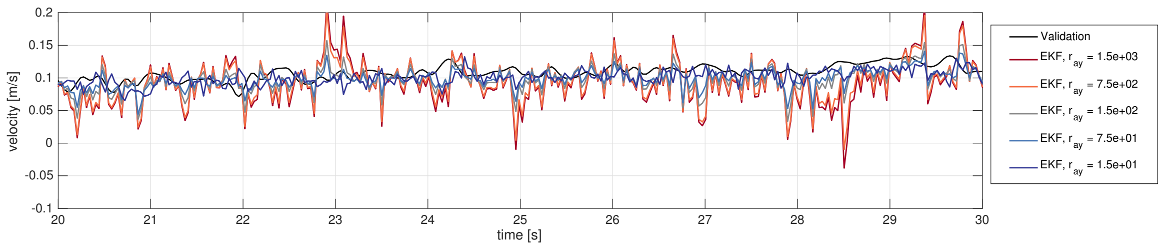
**Figure 6-7:** Longitudinal velocity sensitivity for $r_{enc}$ on two different runs, showing the tradeoff between robustness and accuracy

**Table 6-4:** Performance metrics for sensitivity analysis of $r_{enc}$ tuning on run 3A and 6A

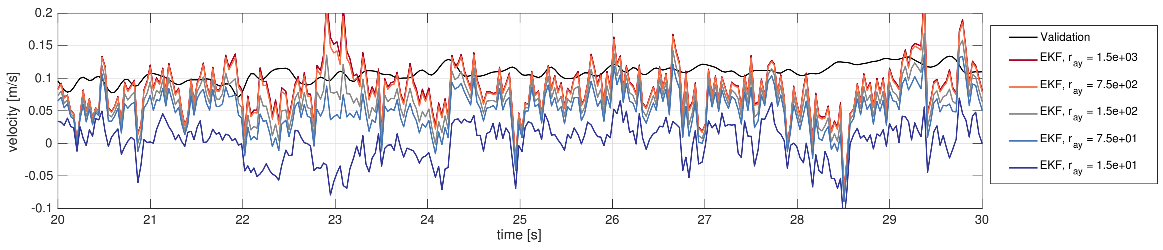
| | Sine steer (3C) | | | | Sine with dwell (6A) | | | |
|---|---|---|---|---|---|---|---|---|
| | Estimated $V_x$ | | Estimated $V_y$ | | Estimated $V_x$ | | Estimated $V_y$ | |
| $r_{enc}$ | RMSE | VAF | RMSE | VAF | RMSE | VAF | RMSE | VAF |
| 0.01 | 0.0632 | 99.05% | 0.0314 | 80.30% | 0.2703 | 93.71% | 0.2028 | 86.25% |
| 0.10 | 0.0578 | 99.14% | 0.0318 | 79.69% | 0.2235 | 95.51% | 0.1872 | 88.32% |
| 0.50 | 0.0591 | 99.00% | 0.0329 | 78.10% | 0.1461 | 97.99% | 0.1536 | 92.14% |
| 1.00 | 0.0691 | 98.81% | 0.0333 | 77.40% | 0.1603 | 97.81% | 0.1527 | 92.19% |
| 10.0 | 0.1186 | 97.67% | 0.0346 | 75.53% | 0.2466 | 96.30% | 0.1576 | 91.50% |

## 6-2-3 Lateral acceleration variance

The next step was to tune the lateral acceleration variance. The calculated lateral acceleration should help the estimation process during (quasi) steady-state maneuvers, when the kinematic model is more prone to the integration of sensor noise and drift. Relying more on the calculated lateral acceleration will however affect robustness. To illustrate this, experiment 4A has been simulated for five values of $r_{a_y}$ and that simulation has been done two times. Once with the correctly identified $C_{\alpha,i}$ $i = \{f, r\}$ and once with it reduced by two thirds. In Figure 6-8a it can be seen that increasing $r_{a_y}$ gives better accuracy when under ideal circumstances. In Figure 6-8b when the cornering stiffness is reduced to 33% of its correct value, a high $r_{a_y}$ pulls down the observer estimate by a lot. Tuning to $r_{a_y} = 150$ gave a good balance between filtering the noisy estimate in normal driving conditions while still maintaining robustness to model uncertainty. This is also evident from the performance metrics in Table 6-5.



**(a)** Lateral velocity sensity for $r_{a_y}$ with correct cornering stiffness



**(b)** Lateral velocity sensitivity for $r_{a_y}$ with 33% of normal cornering stiffness

**Figure 6-8:** Lateral velocity sensitivity for $r_{a_y}$ on part of run 4A, with normal cornering stiffness (top) and 67% reduced cornering stiffness in the calculated lateral acceleration function (bottom)

**Table 6-5:** Performance metrics for sensitivity analysis of $r_{a_y}$ tuning on run 4A for two variations ofr cornering stiffness

| | 100% $C_{\alpha,i}$ (4A) | | | | 33% $C_{\alpha,i}$ (4A) | | | |
|---|---|---|---|---|---|---|---|---|
| | Estimated $V_x$ | | Estimated $V_y$ | | Estimated $V_x$ | | Estimated $V_y$ | |
| $r_{enc}$ | RMSE | VAF | RMSE | VAF | RMSE | VAF | RMSE | VAF |
| $1.5 \cdot 10^3$ | 0.0535 | 92.35% | 0.0373 | 50.95% | 0.0535 | 92.34% | 0.0424 | 40.17% |
| $7.5 \cdot 10^2$ | 0.0536 | 92.34% | 0.0340 | 59.67% | 0.0535 | 92.33% | 0.0430 | 42.62% |
| $1.5 \cdot 10^2$ | 0.0535 | 92.31% | 0.0235 | 81.83% | 0.0535 | 92.26% | 0.0502 | 51.42% |
| $7.5 \cdot 10^1$ | 0.0534 | 92.32% | 0.0199 | 87.44% | 0.0536 | 92.22% | 0.0595 | 51.98% |
| $1.5 \cdot 10^1$ | 0.0531 | 92.30% | 0.0189 | 87.28% | 0.0535 | 92.18% | 0.1011 | 14.63% |

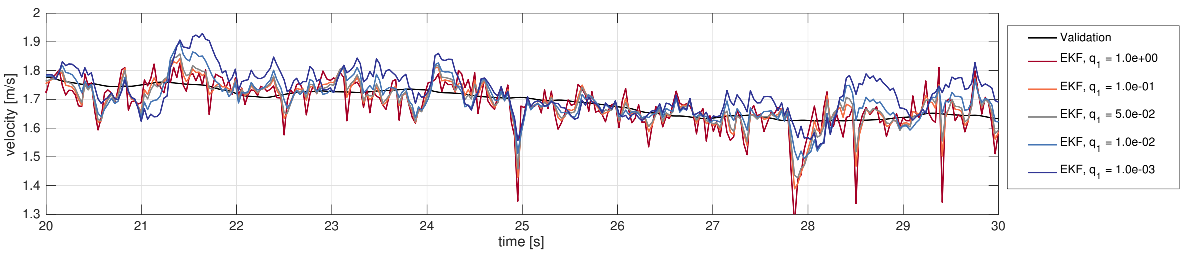
## 6-2-4 Process noise variance

The final parameters that needed to be tuned were the process noise variances. Increasing the process noise variance will have the observer rely more on the correction done by the measurements from the sensors. If the process noise variance is reduced, the observer will filter the measurements more, but can also start integrating signal drift and noise from the IMU. To illustrate this effect, a sensitivity analysis for the process noise variance was done on maneuver 4A. During a quasi steady state maneuver, the risk of integrating IMU errors is clearly visible.

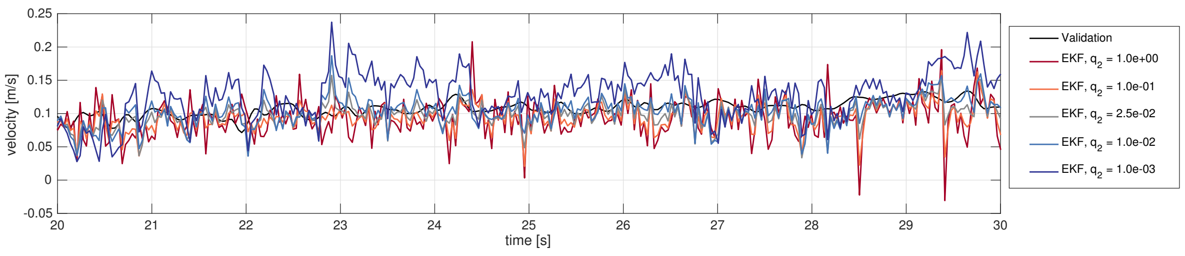
In Figure 6-9a it can be seen that the observer with a low $q_1$ tracks the validation signal roughly, but there are still some spikes from the camera signal propagated in the final estimated. Increasing $q_1$ results in these errors being filtered out. If $q_1$ gets too large, the observer starts integrating errors from the IMU and some oscillations become visible at $t = 21.5$ s and $t = 28$ s. The same observation can be made for $q_2$ in Figure 6-9b. A high $q_2$ propagates more camera noise, but a low $q_2$ starts to give oscillations coming from the IMU drift and noise. Tuning to $q_1 = 0.05, q_2 = 0.025$ gave good results on a variety of runs, which is reflected in the performance metrics in Table 6-6. There was virtually no effect of the tuning of $q_1$ on $V_y$ and $q_2$ on $V_x$.

**Table 6-6:** Performance metrics for sensitivity analysis of process noise tuning on run 4A

| | Estimated $V_x$ | | Estimated $V_y$ | | | Estimated $V_x$ | | Estimated $V_y$ | |
|---|---|---|---|---|---|---|---|---|---|
| $q_1$ | RMSE | VAF | RMSE | VAF | $q_2$ | RMSE | VAF | RMSE | VAF |
| $1.0 \cdot 10^0$ | 0.0660 | 88.40% | 0.0231 | 82.25% | $1.0 \cdot 10^0$ | 0.0540 | 92.28% | 0.0369 | 51.48% |
| $1.0 \cdot 10^{-1}$ | 0.0552 | 91.85% | 0.0233 | 82.05% | $1.0 \cdot 10^{-1}$ | 0.0538 | 92.30% | 0.0277 | 76.20% |
| $5.0 \cdot 10^{-2}$ | 0.0535 | 92.31% | 0.0235 | 81.83% | $2.5 \cdot 10^{-2}$ | 0.0535 | 92.31% | 0.0235 | 81.83% |
| $1.0 \cdot 10^{-2}$ | 0.0556 | 91.58% | 0.0242 | 80.93% | $1.0 \cdot 10^{-2}$ | 0.0535 | 92.23% | 0.0232 | 79.13% |
| $1.0 \cdot 10^{-3}$ | 0.0740 | 85.32% | 0.0256 | 79.49% | $1.0 \cdot 10^{-3}$ | 0.0569 | 91.28% | 0.0460 | 11.51% |



**(a)** Longitudinal velocity sensitivity for $q_1$



**(b)** Lateral velocity sensitivity for $q_2$

**Figure 6-9:** Sensitivity for longitudinal and lateral velocity for changes in process noise on part of run 4A

## 6-2-5 Tuning of other observers

For the other two observer configurations that were simulated, tuning has been performed in roughly the same way. The observer with CVA but without DCE was tuned to have roughly similar performance as the observer with DCE on runs 6A-6C, without sacrificing too much performance on the other runs. If the observer with static covariance is tuned to equal the performance of the observer with DCE on runs 6A-6C, the covariance of the camera measurements must be set to a low value. Because of the low value putting much weight on the camera signals, the performance on the other experiments deteriorates too much. Therefore, a value of 20 for longitudinal and lateral camera noise variance was chosen. This balances performance on linear and nonlinear driving conditions for the observer with the CVA and static covariance.

Exactly this trade-off is why the DCE was implemented in the first place, making this choice obsolete and achieving good performance in both situations. All other settings were kept the same to allow for fair comparisons. For the observer without the CVA the process noise variance was decreased a little and the lateral acceleration variance was decreased by a factor 30 as well, making the observer use the lateral acceleration correction more. This was needed to have good tracking in steady state situations, where the kinematic model is less accurate due to sensor noise and drift.

## 6-3 Results

For all six maneuvers one of the three runs is plotted in this thesis. These results can be seen in Figures 6-10 to 6-15. Next to that the average Root Mean Square Error (RMSE) and Variance Accounted For (VAF) has been calculated for every type of maneuver based on the three repetitions that were available. The average performance can be found in Table 6-7. In this section some observations based on the results will be given.

### 6-3-1 Sine steer performance

When looking at the first maneuver variation out of the three available sine steer variations, maneuver 1A shown in Figure 6-10, it can be seen that the observer with CVA and DCE performs better than the observer without DCE, but still worse than the observer without camera. It estimates the longitudinal velocity better, but still trails behind on the lateral velocity estimation. This is also evident from the performance metrics. It can also be clearly seen that if the camera measurements experience a spike, the observer with DCE is able to handle these sudden errors better than the observer without DCE. Next to that, the observation can be made that the roll angle estimation performs well here, because of the high excitation of lateral acceleration. The estimation of the sideslip angle is comparable between the observer with CVA and DCE and the IMU-encoder-only observer. The static covariance CVA observer deviates more from the validation value.

In maneuver 2C, see Figure 6-11, the rear wheel torque is increased which results in the longitudinal velocity increasing slightly and the peak lateral speed decreasing slightly. The camera signal is less accurate because of the higher speed and the observers using the CVA fall back a bit in performance. As the frequency of the sinusoidal steering input goes from 0.5 Hz to 1.0 Hz in maneuver 3C, see Figure 6-12, the performance of the lateral velocity estimate of the observer claws back a bit. This could be because of the higher frequency of steering input and so higher excitation in lateral dynamics. The kinematic model benefits from this and performance could therefore increase.

### 6-3-2 Ramp steer performance

The next step is assessing the performance of the observers on a quasi steady-state maneuver. This is done on maneuvers 4A-4C and 5A-5C, of which 4A and 5A are shown in this report. In maneuver 4A, shown in Figure 6-13, it can be seen that the DCE adds a good step of performance on the longitudinal and lateral velocity estimate when compared to the observer with CVA but without DCE. The observer without CVA is still superior in estimating the lateral velocity, but falls behind in longitudinal velocity. The superior lateral velocity estimation is probably because the model assumptions of the calculated lateral acceleration are working very well here and the observer without CVA relies heavily on that part. Tire forces are probably not saturated, which allows for a model without combined slip. Next to that there is practically no change in yaw rate, so assuming a steady state on the yaw dynamics is also valid.

When the rear wheel torque is increased in maneuver 5A, the same observations hold up. The observer with CVA and DCE follows closely behind the observer without CVA and the

DCE adds a good step of performance over the observer using static covariance. Again the longitudinal velocity estimate is best for the observer with DCE. To try and keep the plots a little clearer, the camera median has not been plotted in these graphs. Side slip estimation is again best on the IMU-only observer, with the DCE observer following close behind. Roll angle estimation is a bit less on this maneuver, since it is harder to track the small variations in roll angle with a nearly constant lateral acceleration and a very noise roll velocity signal.

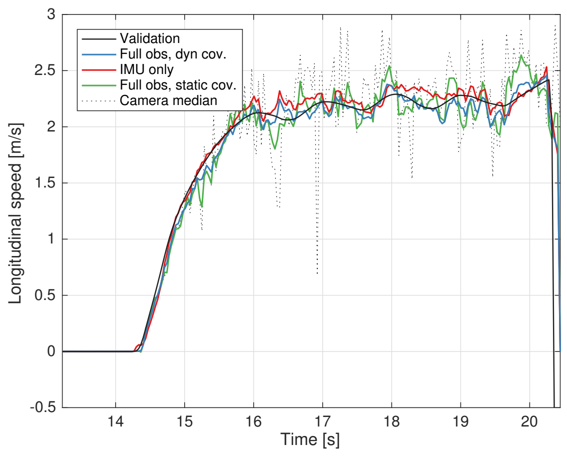
## 6-3-3 Sine with dwell performance

The sine with dwell maneuvers with reduced rear wheel grip give a first look at the areas where adding a camera really makes the difference in comparison to the observer without the CVA. In Figure 6-15 it can be seen that the vehicle speeds up, does a seemingly normal left hand turn and then steers very strongly to the right and holds that steering angle for some time. Because of the rear wheels offering less grip than is expected by the tire model, the calculated lateral acceleration is off by a lot. Next to that, the vehicle reaches a high yaw rate during the counter steer moment of the sine with dwell of up to $6.5\,\mathrm{rad\,s^{-1}}$. This violates the assumption of zero yaw rate acceleration of the calculated lateral acceleration. This results in the observer without CVA delivering poor estimates of lateral velocity between $t = 17\,\mathrm{s}$ and $t = 18.5\,\mathrm{s}$. The observers with CVA do not suffer much from this problem, since the camera signal can do most of the heavy lifting there. It can also be seen that the variance calculated by the DCE scales back exactly at the moments where the camera signal is needed most, between $t = 17\,\mathrm{s}$ and $t = 18.5\,\mathrm{s}$

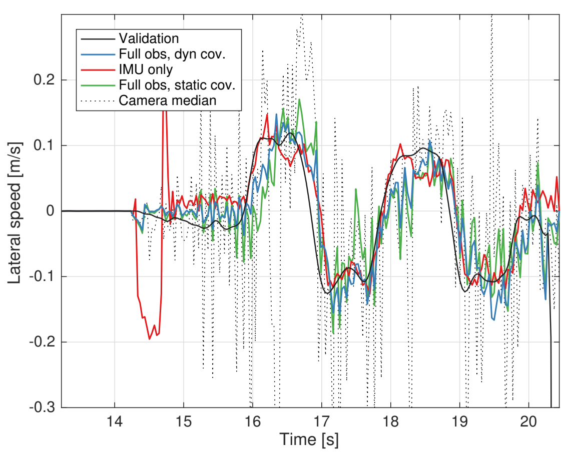
The longitudinal estimation is much more accurate on the observer with DCE. The main reason for this is the vehicle driving backwards for a small moment and the wheel encoder not being able to distinguish a backwards motion from a forward motion. The combination of better longitudinal and lateral velocity estimation results in the observer with DCE and CVA having the best performance at the sideslip angle estimation. The RMSE for the sideslip angle in Table 6-7 looks really large for all observers, but this is mostly because of the $\beta$ becoming really large during the drift.

**Table 6-7:** Average performance metrics all six types of maneuvers. DCE=Dynamic Covariance Estimation, CVA=Computer Vision Algorithm

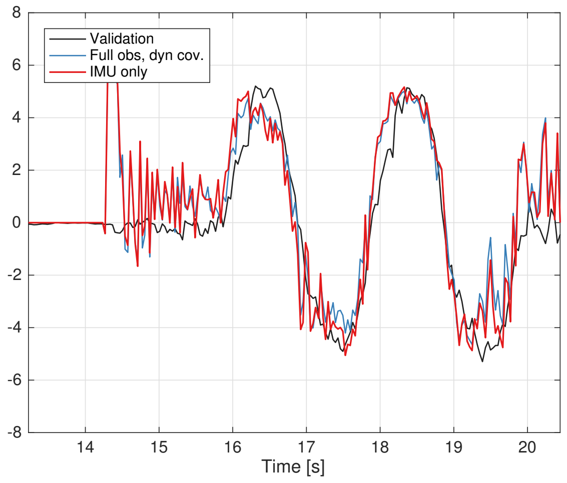
| | Observer type | | Estimated $V_x$ | | Estimated $V_y$ | | Estimated $\beta$ | |
|---|---|---|---|---|---|---|---|---|
| | DCE | CVA | RMSE | VAF | RMSE | VAF | RMSE | VAF |
| 1A-1C | Yes | Yes | 0.0685 | 99.26% | 0.0458 | 63.64% | 1.2598 | 67.43% |
| | No | No | 0.0787 | 99.28% | 0.0378 | 79.98% | 0.9985 | 81.49% |
| | Yes | No | 0.1423 | 96.86% | 0.0674 | 21.64% | 1.8255 | 30.89% |
| 2A-2C | Yes | Yes | 0.0894 | 98.96% | 0.0535 | 51.26% | 1.4623 | 52.40% |
| | No | No | 0.0788 | 98.87% | 0.0469 | 70.55% | 1.2027 | 72.69% |
| | Yes | No | 0.1913 | 96.58% | 0.0708 | 12.03% | 1.9844 | 11.17% |
| 3A-3C | Yes | Yes | 0.0798 | 99.26% | 0.0496 | 55.31% | 1.3565 | 60.62% |
| | No | No | 0.0766 | 99.22% | 0.0415 | 73.09% | 1.0882 | 74.85% |
| | Yes | No | 0.1574 | 96.80% | 0.0738 | -2.48% | 2.0174 | 5.99% |
| 4A-4C | Yes | Yes | 0.0639 | 90.60% | 0.0241 | 78.21% | 0.9314 | 71.66% |
| | No | No | 0.0795 | 91.14% | 0.0191 | 85.07% | 0.8014 | 82.44% |
| | Yes | No | 0.1002 | 82.68% | 0.0355 | 57.30% | 1.3264 | 45.70% |
| 5A-5C | Yes | Yes | 0.0727 | 96.86% | 0.0331 | 38.79% | 1.0236 | 14.62% |
| | No | No | 0.1016 | 97.08% | 0.0242 | 71.87% | 0.6479 | 81.49% |
| | Yes | No | 0.1612 | 93.20% | 0.0480 | -9.80% | 1.5497 | -78.38% |
| 6A-6C | Yes | Yes | 0.1533 | 97.99% | 0.1895 | 85.53% | 14.1895 | 89.88% |
| | No | No | 0.2889 | 94.82% | 0.3281 | 63.49% | 25.8612 | 71.55% |
| | Yes | No | 0.2522 | 95.81% | 0.2234 | 78.18% | 16.1103 | 85.76% |



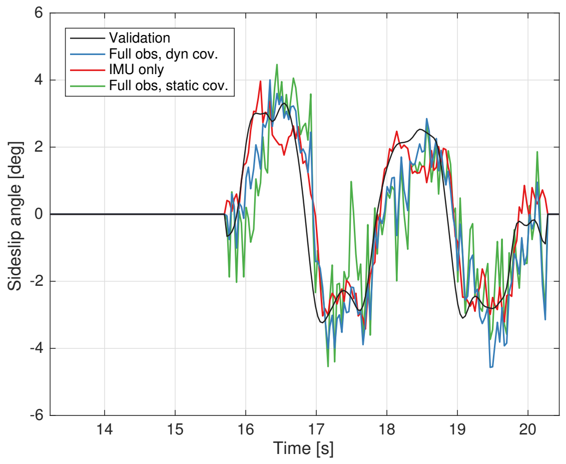
**(a)** Longitudinal speed



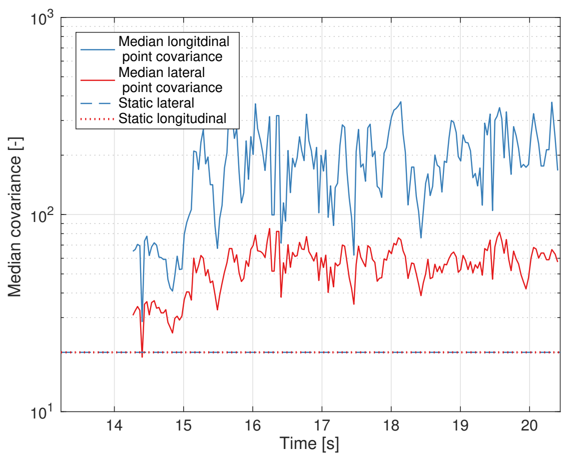
**(b)** Lateral speed



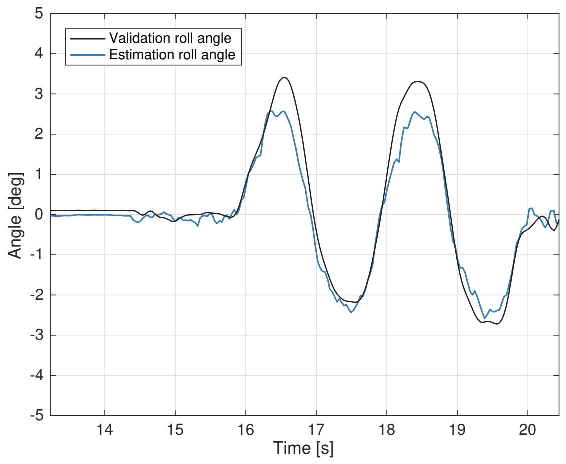
**(c)** Lateral acceleration



**(d)** Sideslip angle

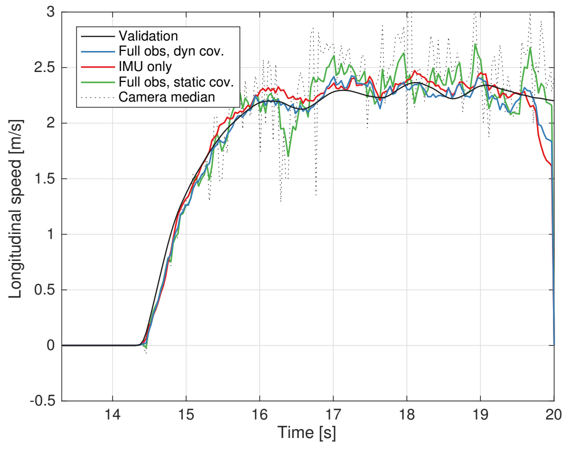


**(e)** Median dynamic covariance

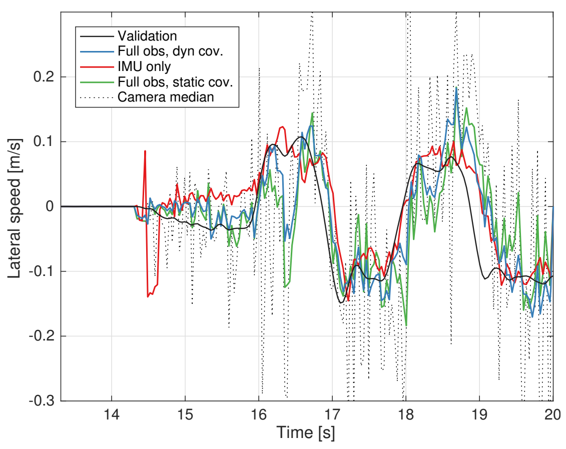


**(f)** Roll angle (top) and pitch angle (bottom)

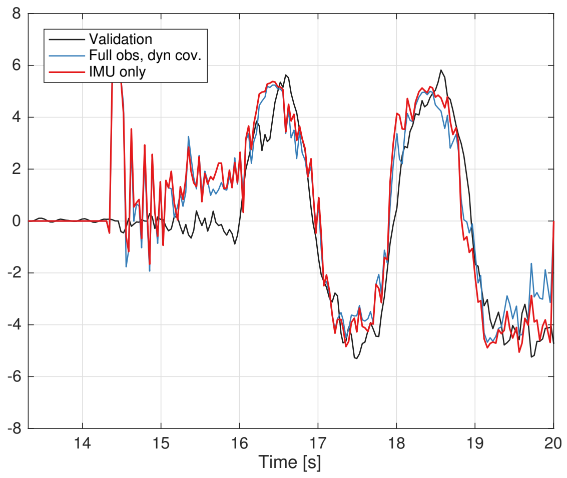
**Figure 6-10:** Results for maneuver 1A, sine steer, $f = 0.5\,\mathrm{Hz}, \delta_f = [-13°, 13°], F_{xR} = 1.7N$



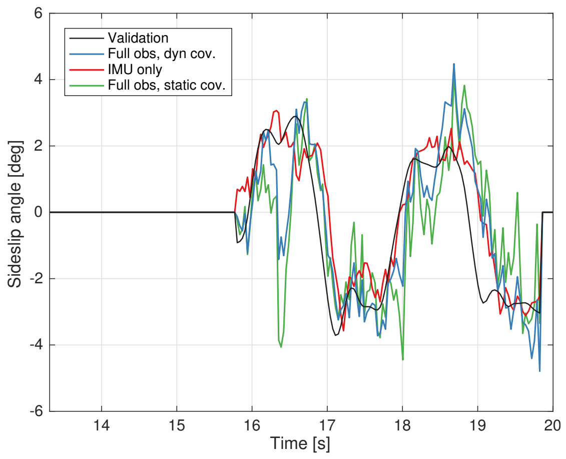
**(a)** Longitudinal speed



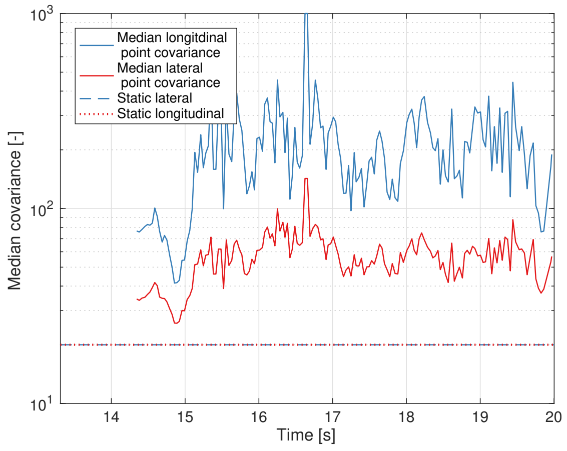
**(b)** Lateral speed



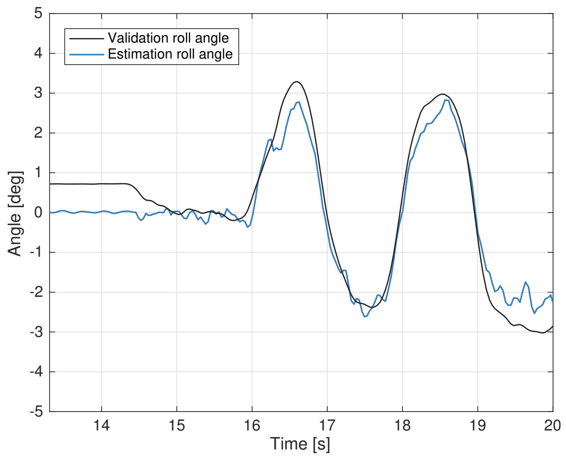
**(c)** Lateral acceleration



**(d)** Sideslip angle

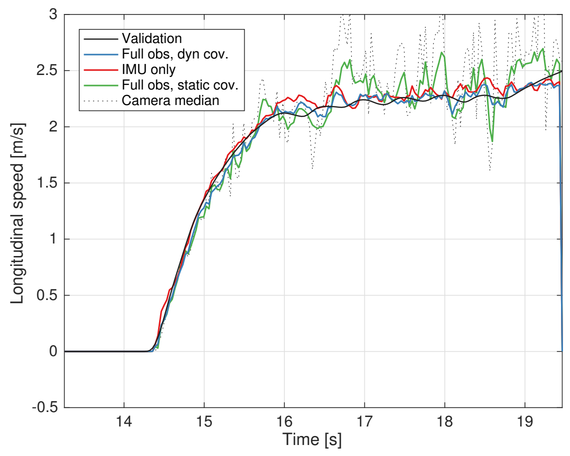


**(e)** Median dynamic covariance

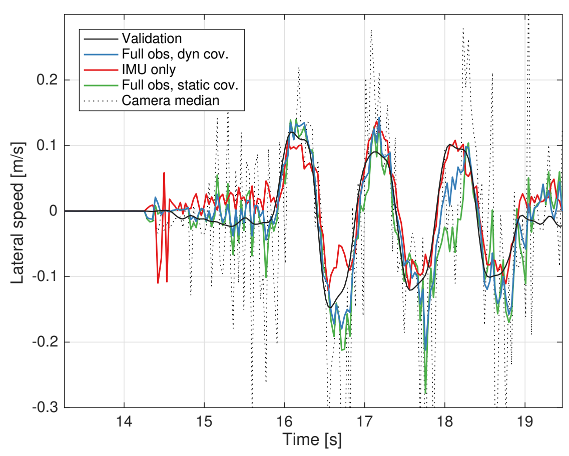


**(f)** Roll angle (top) and pitch angle (bottom)

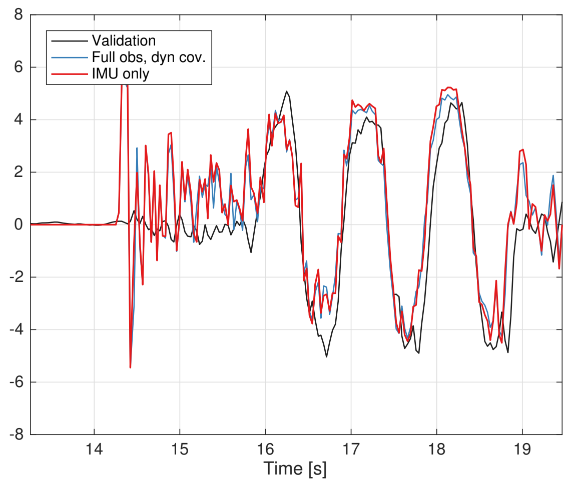
**Figure 6-11:** Results for maneuver 2C, sine steer, $f = 0.5\,\text{Hz}, \delta_f = [-13^\circ, 13^\circ], F_{xR} = 2.2N$



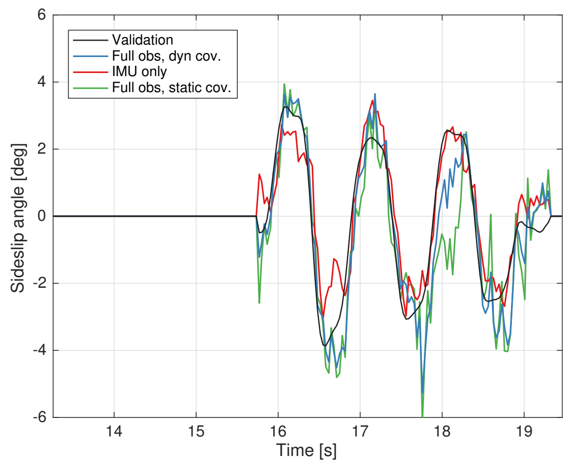
**(a)** Longitudinal speed



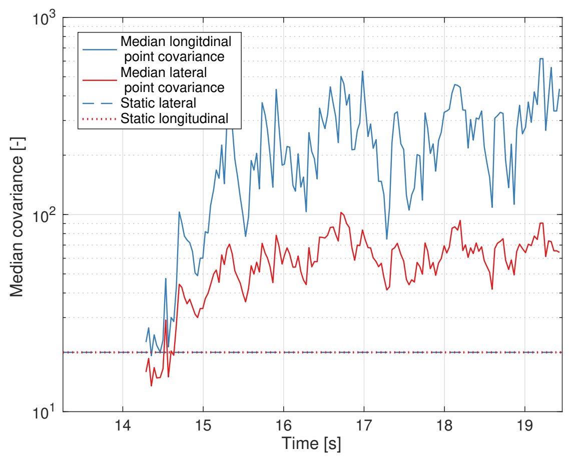
**(b)** Lateral speed



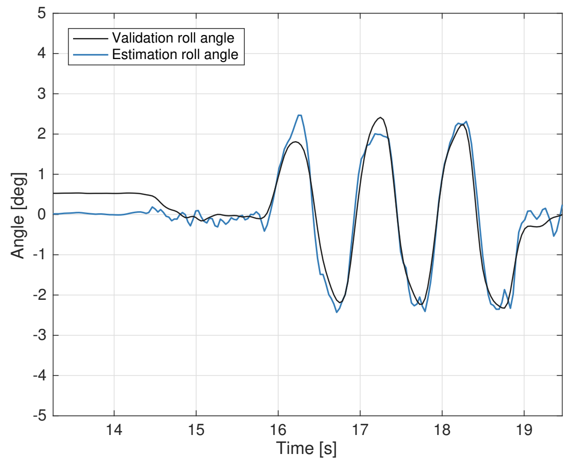
**(c)** Lateral acceleration



**(d)** Sideslip angle



**(e)** Median dynamic covariance



**(f)** Roll angle (top) and pitch angle (bottom)

**Figure 6-12:** Results for maneuver 3C, sine steer, $f = 1.0\,\mathrm{Hz}, \delta_f = [-13^\circ, 13^\circ], F_{xR} = 2.2N$

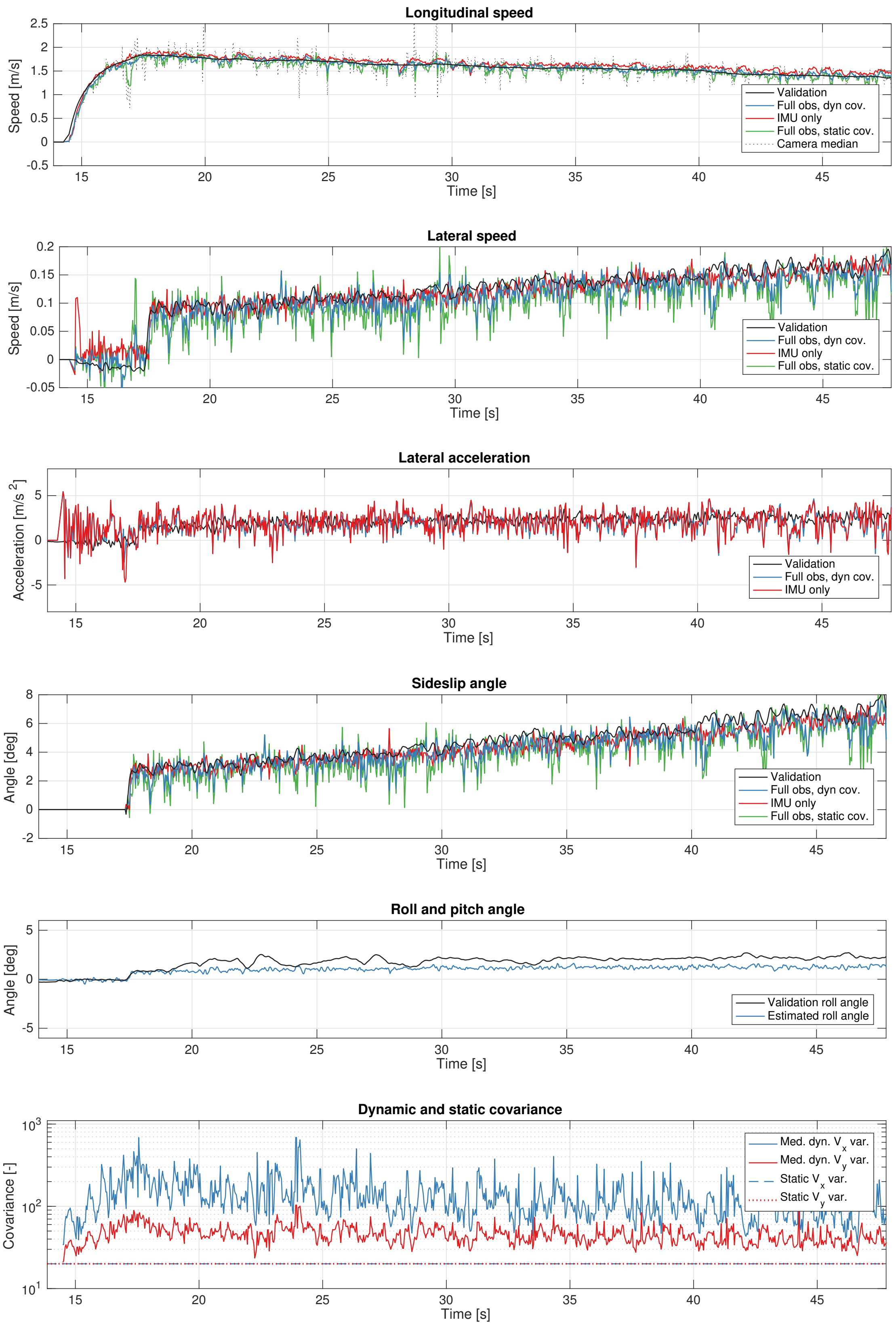


**Figure 6-13:** Results for maneuver 4A, $\Delta\delta_f/\Delta t = 0.3°/s, \delta_f = [7.5, 16.5], F_{xR} = 1.7N$

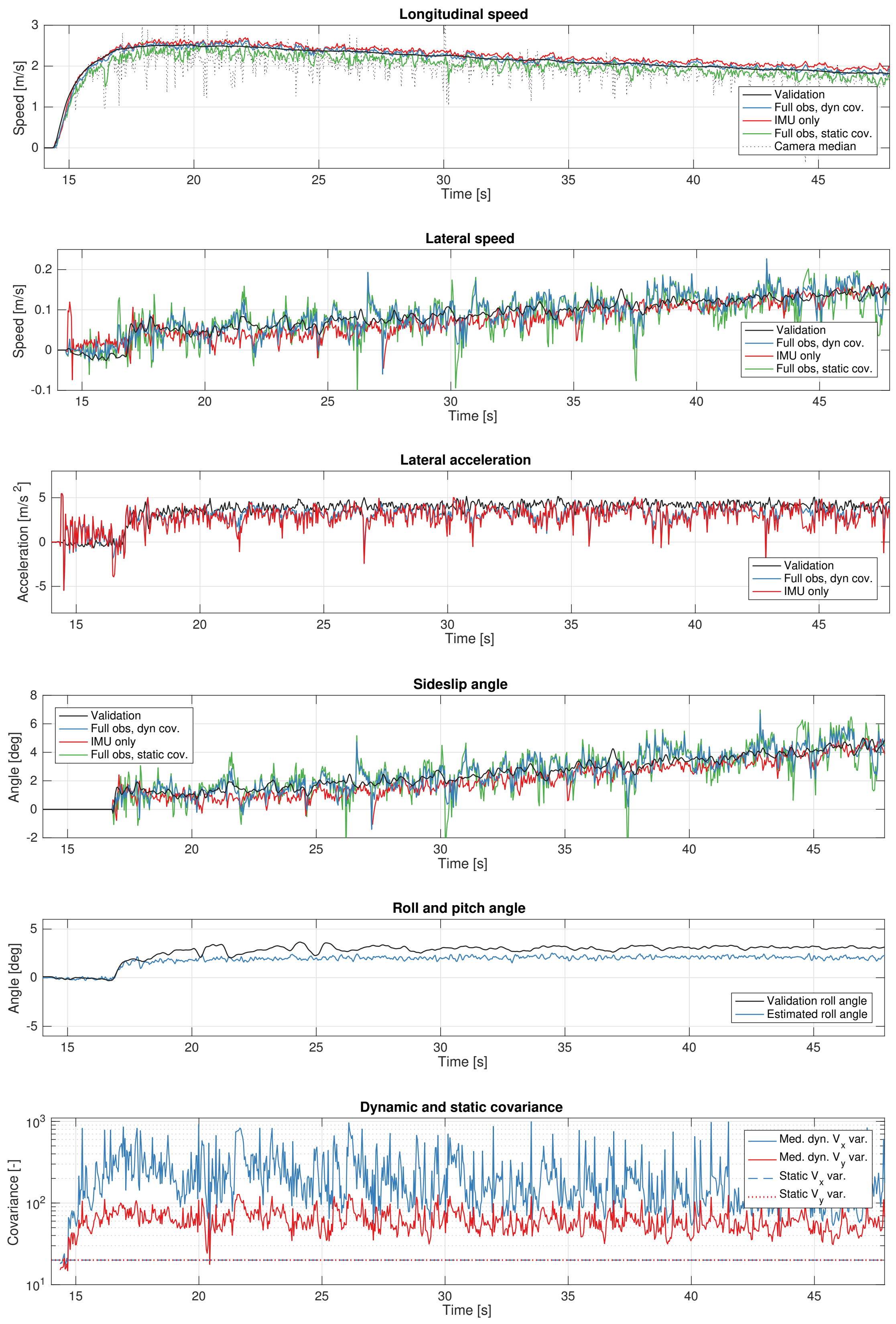
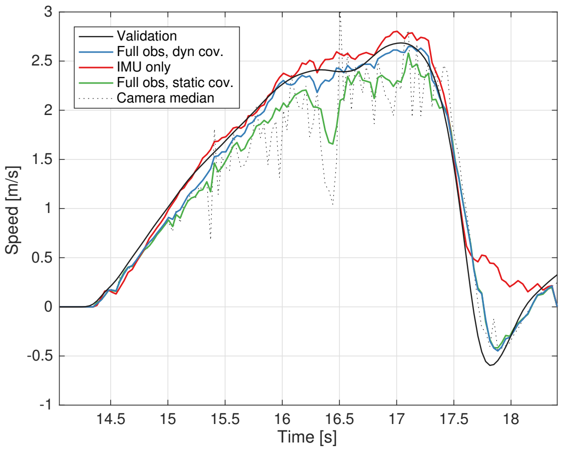
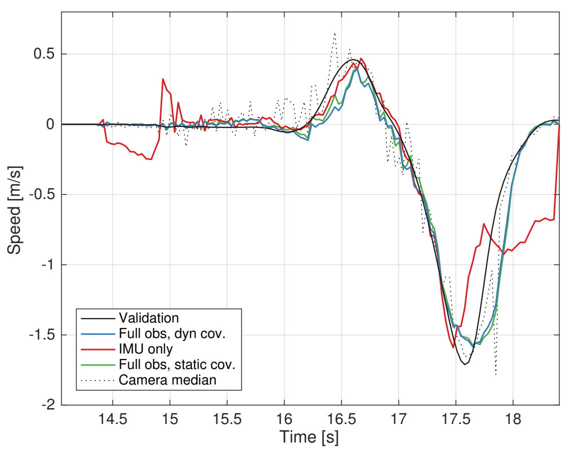


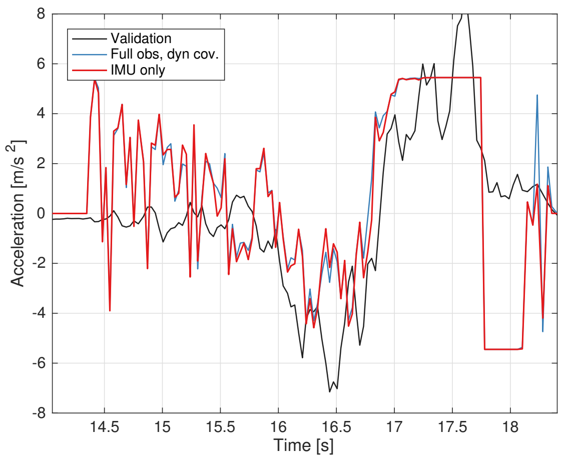
**Figure 6-14:** Results for maneuver 5A, $\Delta\delta_f/\Delta t = 0.3°/s, \delta_f[7.5, 16.5], F_{xR} = 2.2N$



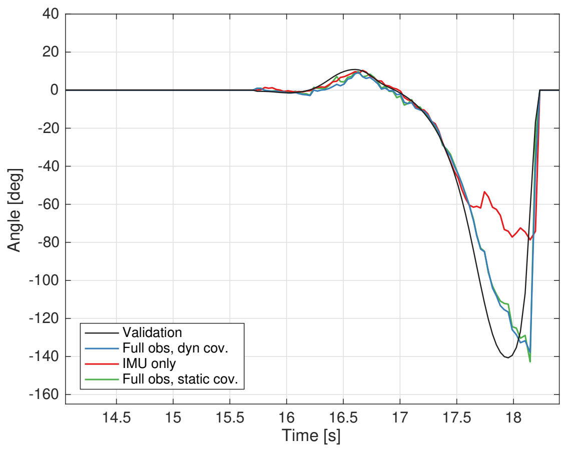
**(a)** Longitudinal speed



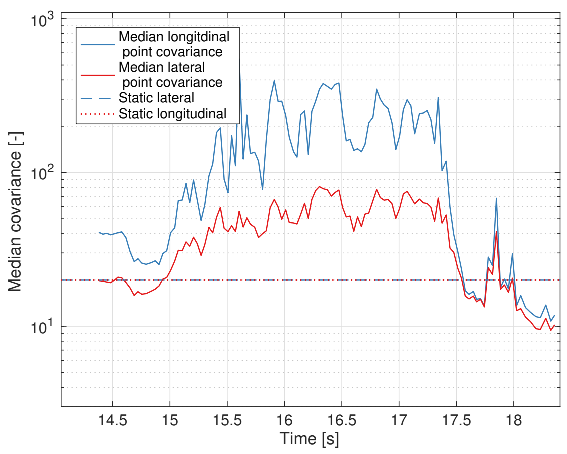
**(b)** Lateral speed



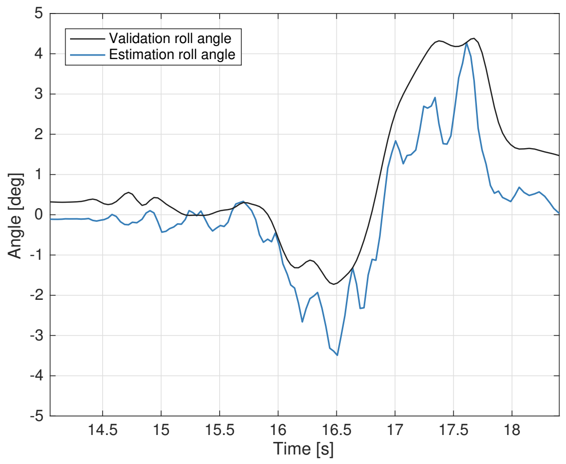
**(c)** Lateral acceleration



**(d)** Sideslip angle



**(e)** Roll angle



**(f)** Median adaptive covariance for longitudinal and lateral tracked points

**Figure 6-15:** Experiment 6A, sine with dwell at $f_\delta = 0.7\,\mathrm{Hz}, \delta_f = [-13°, 13°], F_{xR} = 2.7N$, reduced rear wheel grip

## 6-4 Robustness

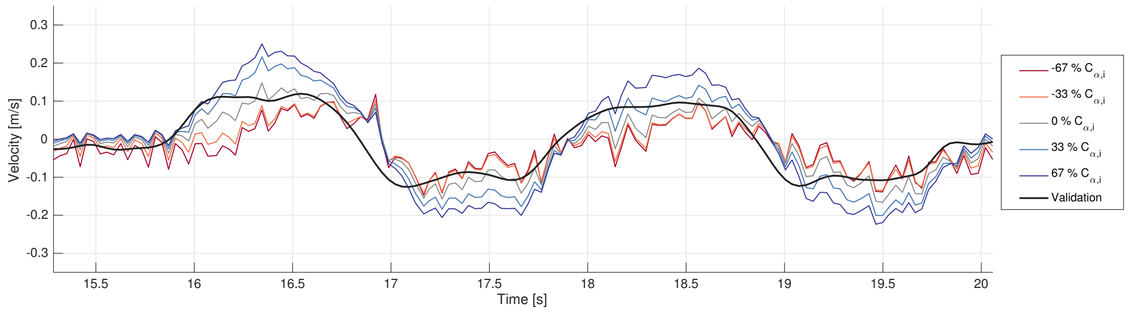
To test the robustness of the newly developed methodology to model uncertainty, several of the previously shown maneuvers are run with different model parameters. Two sine steer maneuvers and two ramp steer maneuvers are run with five different variations of tire parameters, where the tire stiffness parameters $C_{\alpha,i}, i = \{f, r\}$ were adjusted in steps of $1/3^{\mathrm{rd}}$ with respect to the original identified value. Only the effect on lateral velocity estimate is discussed in this section, since the effect on longitudinal speed is negligible. No real physical properties of the vehicle or tires were altered for these experiments, the tire model was only altered digitally in the algorithm.

For maneuver 2A, the first of the two sine steer maneuvers, it can be seen in Figure 6-16 that if the tire parameters are scaled down, the estimate from the observer with CVA and DCE scales down a bit. For the observer without CVA, the estimate completely changes sign. Here the improvement in robustness from adding a camera to the system starts to surface. For maneuver 3C the frequency of the sinusoidal steering input is doubled. In Figure 6-17 it can be seen that the estimates produced by both observers is a little less sensitive to variations in tire parameters. This is probably an effect of having more excitation in the lateral dynamics and therefore better estimation from the kinematic model. This is also backed up by the performance metrics in Table 6-8.

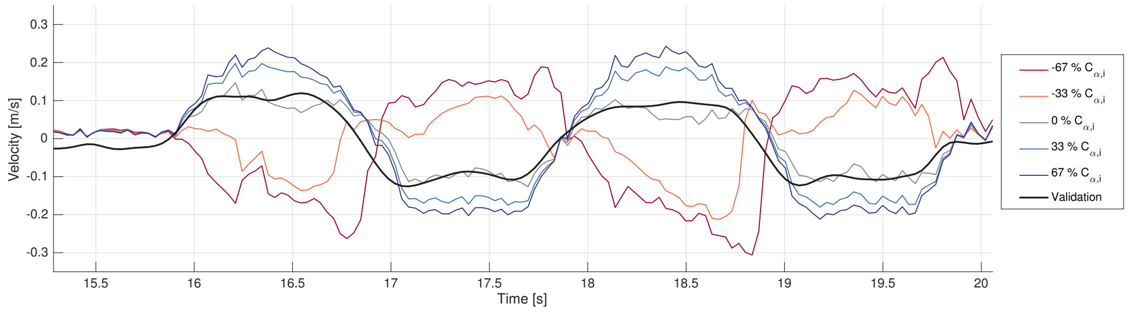
For ramp steer maneuver 4A the results can be seen in Figure 6-18. There, the benefit of adding a camera to the system is clear again, especially when regarding the performance metrics in Table 6-8. The RMSE of the estimate of the observer with camera increases by 114% when stiffness parameters are scaled down by 66%. However, the RMSE of the estimate of the observer without camera increases by 1403%. The same effect can be observed even stronger for maneuver 5A in Figure 6-19. Estimates of lateral velocity of the observer without camera go down to values of $V_y = -0.5\,\mathrm{m\,s^{-1}}$ when tire parameters are scaled down by 66%, while the true value of the validation speed is in the range of $V_y = 0.05 - 0.15\,\mathrm{m\,s^{-1}}$. The estimate of the observer with camera is again much more accurate and closer to the validation value.

**Table 6-8:** Performance benchmarks for run 4A and 5A with varying tire stiffness parameters

| | | CVA & DCE estimated $V_y$ | | Without CVA estimated $V_y$ | | | CVA & DCE estimated $V_y$ | | Without CVA estimated $V_y$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $\Delta C_{\alpha,i}$ | RMSE | VAF | RMSE | VAF | $\Delta C_{\alpha,i}$ | RMSE | VAF | RMSE | VAF |
| | *Exp. 2A* | | | | | *Exp. 3C* | | | | |
| Sinesteer | -66% | 0.0590 | 48.20% | 0.2099 | -300.00% | -66% | 0.0452 | 71.52% | 0.1791 | -300.00% |
| Sinesteer | -33% | 0.0514 | 59.18% | 0.1471 | -239.51% | -33% | 0.0368 | 77.04% | 0.1184 | -173.50% |
| Sinesteer | 0% | 0.0405 | 74.20% | 0.0344 | 83.27% | 0% | 0.0333 | 78.70% | 0.0332 | 82.80% |
| Sinesteer | 33% | 0.0480 | 63.87% | 0.0552 | 55.80% | 33% | 0.0413 | 66.03% | 0.0524 | 49.40% |
| Sinesteer | 66% | 0.0642 | 35.84% | 0.0753 | 15.02% | 66% | 0.0559 | 38.29% | 0.0723 | -0.36% |
| | *Exp. 4A* | | | | | *Exp. 5A* | | | | |
| Ramp steer | -66% | 0.0503 | 50.04% | 0.2631 | -300.00% | -66% | 0.0456 | -32.29% | 0.3358 | -300.00% |
| Ramp steer | -33% | 0.0387 | 67.74% | 0.0491 | 65.49% | -33% | 0.0405 | -1.89% | 0.2716 | -300.00% |
| Ramp steer | 0% | 0.0235 | 81.34% | 0.0175 | 86.80% | 0% | 0.0268 | 30.99% | 0.0251 | 60.13% |
| Ramp steer | 33% | 0.0161 | 88.48% | 0.0194 | 89.25% | 33% | 0.0434 | 53.00% | 0.0478 | 69.05% |
| Ramp steer | 66% | 0.0165 | 91.87% | 0.0250 | 88.96% | 66% | 0.0654 | 63.48% | 0.0733 | 66.23% |

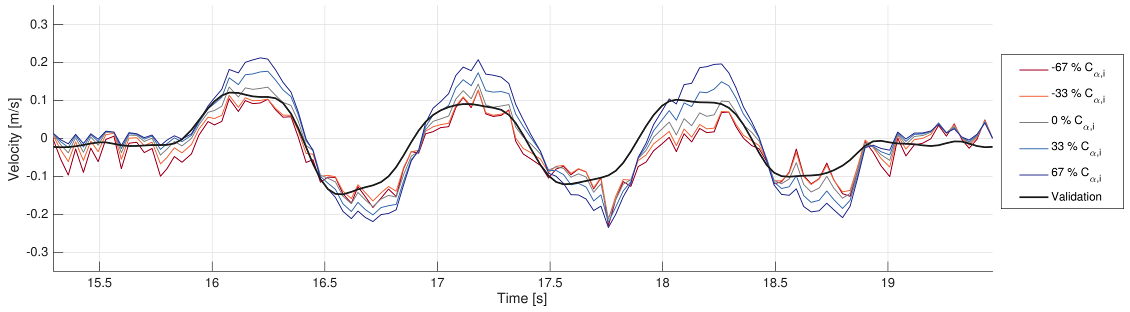


**(a)** Lateral velocity sensitivity for observer with camera and DCE

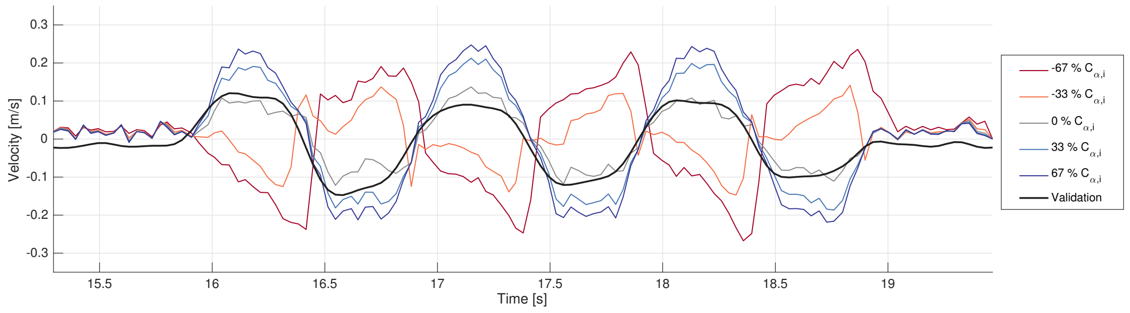


**(b)** Lateral velocity sensitivity for observer with only the IMU

**Figure 6-16:** Lateral velocity sensitivity for changes in the front and rear cornering stiffness parameter in the calculated lateral acceleration function performed on experiment 2A

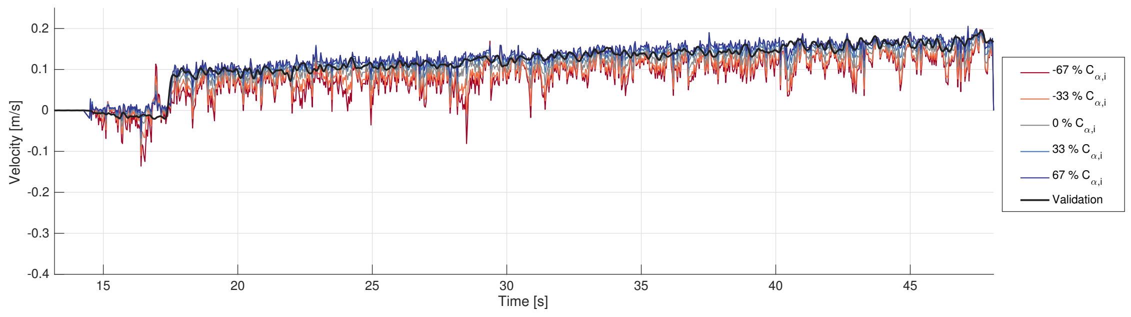


**(a)** Lateral velocity sensitivity for observer with camera and DCE

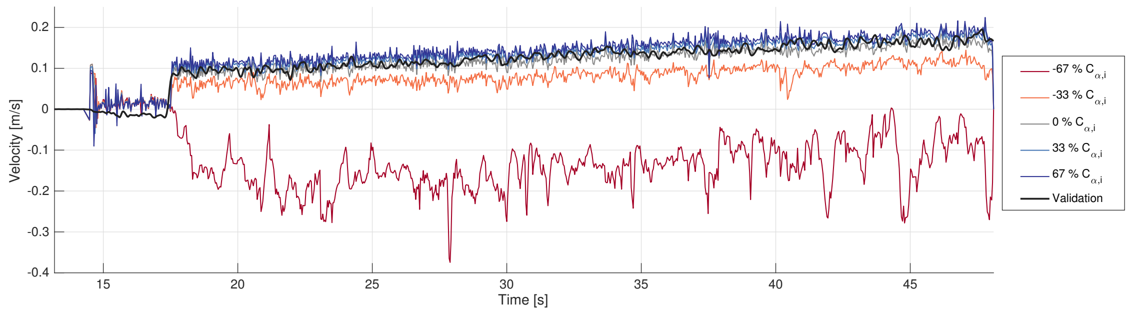


**(b)** Lateral velocity sensitivity for observer with only the IMU

**Figure 6-17:** Lateral velocity sensitivity for changes in the front and rear cornering stiffness parameter in the calculated lateral acceleration function performed on experiment 3C

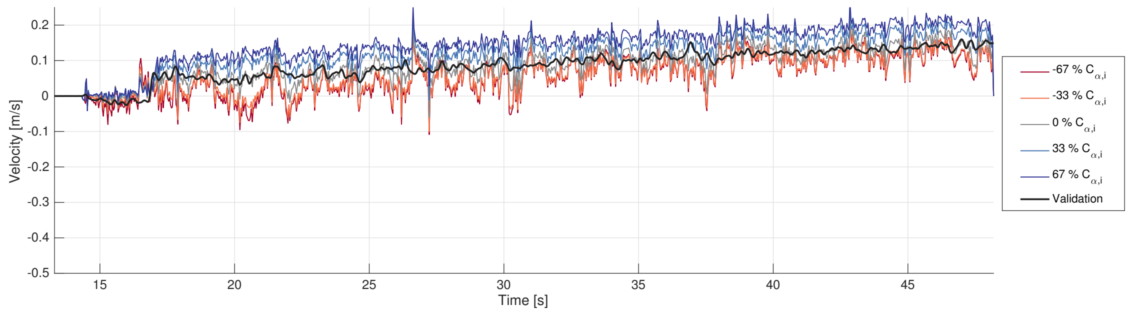


**(a)** Lateral velocity sensitivity for observer with camera and DCE

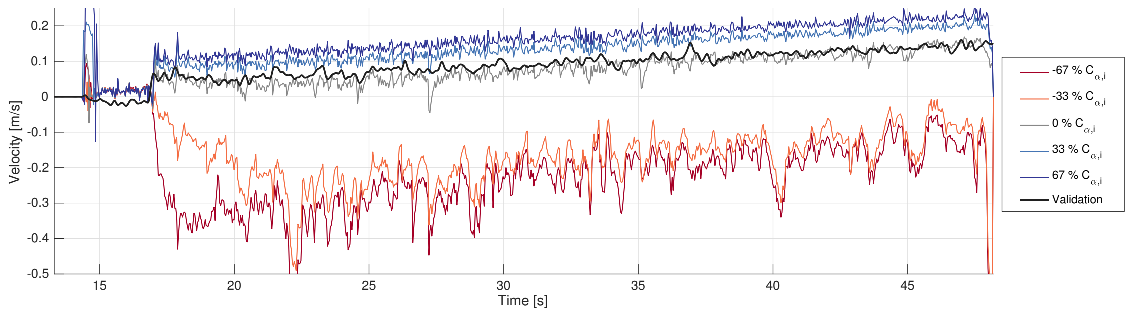


**(b)** Lateral velocity sensitivity for observer with only the IMU

**Figure 6-18:** Lateral velocity sensitivity for changes in the front and rear cornering stiffness parameter in the calculated lateral acceleration function performed on experiment 4A



**(a)** Lateral velocity sensitivity for observer with camera and DCE



**(b)** Lateral velocity sensitivity for observer with only the IMU

**Figure 6-19:** Lateral velocity sensitivity for changes in the front and rear cornering stiffness parameter in the calculated lateral acceleration function performed on experiment 5A

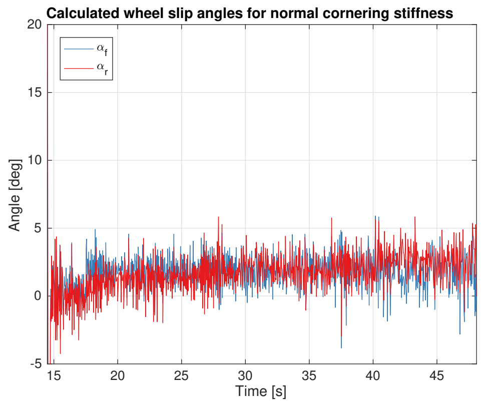
An interesting observation in all maneuvers is that the lateral velocity estimate from the observer without a camera becomes negative if the tire parameters are scaled down by a lot. An explanation for this can be found by looking at the wheel slip angles $\alpha_f, \alpha_r$. In Figure 6-20 the wheel slip angles calculated by the observer without camera are shown for maneuver 4A. Once with the correctly identified tire parameters and once with the tire parameters digitally scaled down in the observer by 66%. Both observers want to track a lateral acceleration of 2.5-3.5 $\mathrm{m\,s^{-2}}$. The lateral acceleration for maneuver 4A was already shown in Figure 6-13. For the observer with normal tire parameters nothing strange happens and the wheel slip angles look normal. However, for the observer with reduced parameters the wheel slip angles are very large. The observer still tries to reach a lateral acceleration of 2.5-3.5 $\mathrm{m\,s^{-2}}$. The lateral acceleration was calculated using:

$$a_y = \frac{1}{m}\left(F_{yf}\cos\delta_f + F_{yr}\right) \tag{6-2}$$

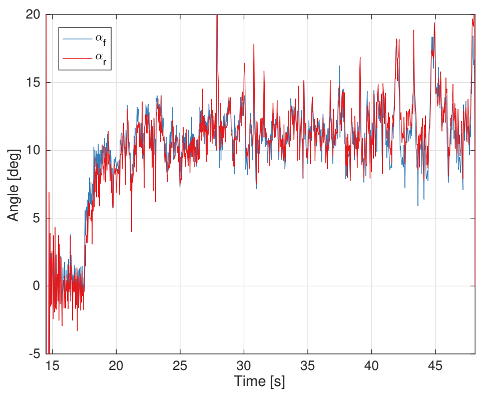
$$F_{yi} = \frac{C_i}{k_i}\tanh k_i\alpha_i \quad i = \{f, r\} \tag{6-3}$$

$$\alpha_f = \left(\delta_f - \frac{V_y + l_f\omega_z}{V_x}\right) \quad \alpha_r = \left(-\frac{V_y - l_r\omega_z}{V_x}\right) \tag{6-4}$$

This means that the only way to reach the same lateral acceleration in (6-2) is to make the tire forces have the same value as they would with normal parameters. In (6-3) it can be seen that if $C_i$ is scaled down, $\alpha_i$ needs to increase. Finally it follows from (6-4) that to increase $\alpha_f$ and $\alpha_r$, $V_y$ needs to decrease. This is a result of the other parameters either being constant or measured using other sensors. This explains why in the case where the tire parameters are scaled down a lot, the lateral velocity becomes negative.



**(a)** Wheel slip angles for correctly identified tire parameters

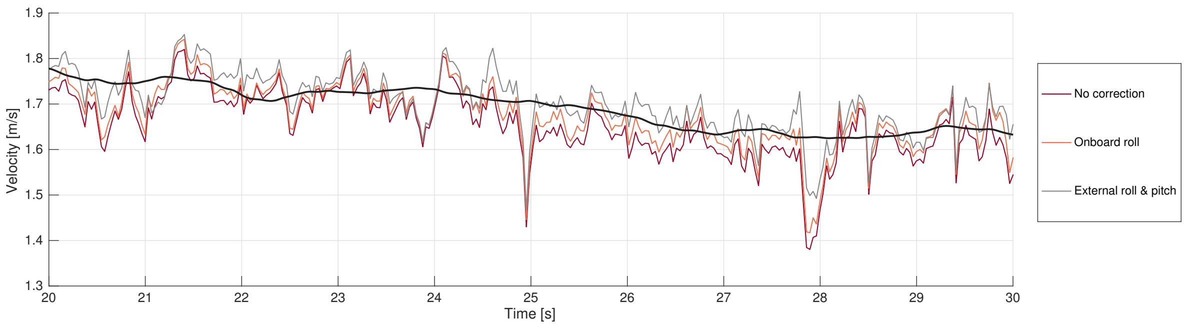


**(b)** Wheel slip angles for 67% reduced tire parameters in the EKF

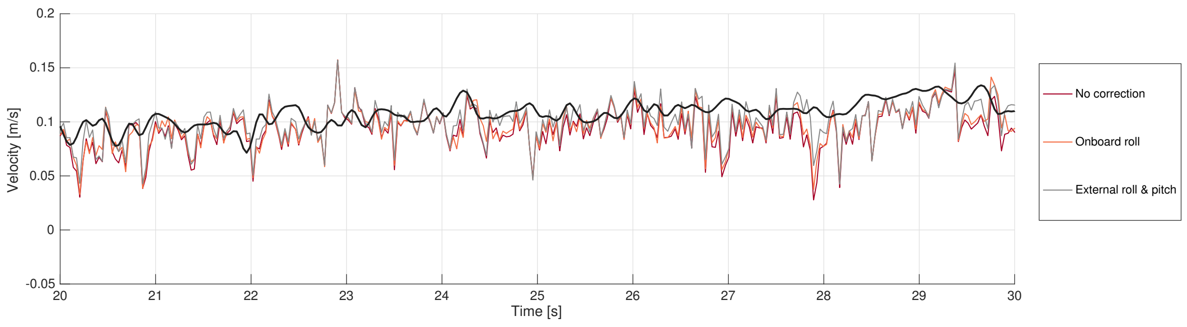
**Figure 6-20:** Longitudinal velocity sensitivity for $r_{enc}$ on two different runs, showing the tradeoff between robustness and accuracy

## 6-5 Roll correction

During all experiments presented above, the roll correction on the homography was performed using the estimate coming from the roll angle observer. On dynamic maneuvers the roll angle estimation performance was decent, however on steady state maneuvers it could still be improved. To assess the effectiveness of the roll angle estimation an evaluation was done on maneuver 4A. Three version of the observer with CVA and DCE were run. Once without any correction for camera pose, once with only the onboard roll observer and once with a correction done by the measured roll and pitch angles from the MCS. In Figure 6-21 it can be seen that the longitudinal velocity estimate is much closer when pitch correction is also done. For the lateral velocity the observer with pitch and roll correction based on MCS angles doesn't seem to be much better in the plots, but when looking at the performance metrics in Table 6-9 it can be seen that it still outperforms the observers without pitch correction by a margin. However, just applying roll correction based on the onboard angles already gives a sizable improvement over the observer without any camera pose correction. Finally, these results show how much potential improvement is possible when a proper roll and pitch observer would be implemented.



**(a)** Longitudinal velocity estimate from EKF with different versions of camera pose correction



**(b)** Lateral velocity estimate from EKF with different versions of camera pose correction

**Figure 6-21:** Longitudinal and lateral velocity from observer with CVA and DCE for different versions of camera pose correction

**Table 6-9:** Performance metrics for the observer with CVA and DCE on maneuver 4A with different versions of pitch and roll correction

| | Correction for: | | Estimated $V_x$ | | Estimated $V_y$ | | Estimated $\beta$ | |
|---|---|---|---|---|---|---|---|---|
| | Pitch | Roll | RMSE | VAF | RMSE | VAF | RMSE | VAF |
| Exp. 4A | No | No | 0.0642 | 95.37% | 0.0289 | 75.62% | 0.9852 | 70.35% |
| | No | Yes, onboard | 0.0492 | 94.81% | 0.0248 | 79.06% | 0.8829 | 74.56% |
| | Yes, MCS | Yes, MCS | 0.0476 | 94.77% | 0.0209 | 81.83% | 0.7748 | 77.07% |

## 6-6 Conclusion

In this chapter experiments were presented that showed the performance of the proposed methodology in comparison to the same observer without DCE and that same observer without the camera. In normal driving conditions the observer without camera still prevails. During highly dynamic maneuvers the kinematic part of the observer still works well, while during steady state driving the calculated lateral acceleration works well. The observer with CVA and DCE has an increase in RMSE of 25% and a relative decrease in VAF of 25% in lateral velocity estimation performance when compared to the observer without camera. This is an average over all 15 normal driving experiments. However, when comparing both observers with CVA the observer without DCE has an increase of 44% in RMSE and a relative decrease in VAF of 80% in lateral velocity estimation when compared to the observer with DCE on all 15 maneuvers. This shows that using the DCE does indeed improve estimates.

On average, the longitudinal velocity estimation is slightly better for the observer with DCE than for the observer without camera, with a reduction of 10% in RMSE when adding CVA and DCE to the system averaged over all 15 experiments. Final sideslip angle estimation performance for the observer with CVA and DCE is close to the estimation performance of the observer without camera.

During experiments where the vehicle drifted due to altered rear grip, maneuvers 6A-6C, both observers with camera proved to be much better. When comparing the observer with CVA and DCE to the observer without camera, the observer with CVA and DCE has a reduction in RMSE of 42% and a relative increase in VAF of 35%. The robustness also shows when analyzing the sensitivity to wrongly identified tire model parameters. The observer with camera has an average increase of 66% on the RMSE of the lateral velocity estimation, where the observer without camera has an increase of 900% in RMSE when the cornering stiffness parameter is set to 33% of its estimated value. The average is calculated over all four simulated maneuvers. This shows that having a camera can greatly improve results when the tire friction estimation is not fully up to date.

Finally, it was shown that implementing a roll angle observer and correcting the camera pose for vehicle body roll does improve results. Improving the performance of the roll angle observer and implementing a pitch angle observer will improve results, as was shown by using the externally measured angles from the MCS for camera pose correction.

Chapter 7

# Conclusions and Recommendations

In this chapter conclusions and recommendations will be presented. First, conclusions will be drawn on the proposed methodology based on the experiments conducted on the test vehicle. Then recommendations will be proposed that could be taken into account when continuing work on this observer methodology.

## 7-1 Conclusions

Controllers that can let a vehicle drift autonomously still remain as one of the most promising ideas in vehicle dynamics control. An important step in bringing these controllers to consumer vehicles is feasible estimation of the body sideslip angle. Current techniques rely on expensive sensor setups or an exhaustive knowledge of vehicle parameters. In this project an observer using only cost efficient sensors has been proposed.

### 7-1-1 Observer architecture

At the core of the proposed technique lies a hybrid kinematic-dynamic model. The model takes the kinematic equations for longitudinal and lateral velocity as a basis. As an output the lateral acceleration is calculated based on a single track model with a pure lateral force hyperbolic tangent tire model. This is combined with the measurements from an Inertial Measurement Unit (IMU), wheel speed sensor and velocity measurement from the camera in an Extended Kalman Filter (EKF). Finally, a separate observer estimates the vehicle roll angle that is needed for the camera to estimate the vehicle velocity. In this way a compact yet effective observer could be created that can take advantage of the best of both models.

The camera sees the ground plane and estimates the vehicle body speed by following points on the ground plane. This is done by using a Computer Vision Algorithm (CVA) that employs Shi-Tomasi corner detection and an Iterative Pyramidal implementation of Lucas-Kanade optical flow. Because the camera velocity measurements contained too much noise to be used directly in the EKF, the covariance matrix of the measurement noise is updated online by the Dynamic Covariance Estimation (DCE). Based on a model of the error of the camera measurements the algorithm adjusts the covariance for each tracked point based on point location, tracking quality and vehicle states.

### 7-1-2 Observer performance

To validate the observer performance, open loop maneuvers were performed on a 1:10 scale vehicle, the Berkeley Autonomous Race Car (BARC). All sensor data of the BARC was recorded and later used to simulate the developed methodology as if it would run online. During the experiments the vehicle location was tracked using a Motion Capture System (MCS), so ground truth data was available for experiment validation. During experiments the newly developed methodology with CVA and DCE was validated against an observer without DCE but with CVA and an observer without a camera at all.

In linear driving conditions under correctly identified model parameters the observer without a camera still performs better than observer with a camera. Averaged over 15 maneuvers, 9 dynamic and 6 quasi-steady state, the observer with CVA and DCE was slightly better in longitudinal velocity estimation (-9% RMSE) and worse in lateral velocity estimation (+25% RMSE) and body sideslip angle estimation (+29% RMSE). Roughly the same observations were true for the VAF. When comparing the observers with CVA, the observer with DCE was a lot better in estimating all three quantities. The RMSE increased respectively 99% for longitudinal velocity estimation, 44% for lateral velocity estimation and 45% for sideslip angle estimation when removing the DCE from the system.

An explanation for the superior performance of the observer without camera in normal driving conditions can be found in the way the observer is set up. During quasi-steady state driving conditions, for example ramp steer maneuvers, the model assumptions of the calculated lateral acceleration are valid and correction based on the measured lateral acceleration works very well. During maneuvers where the lateral dynamics were excited more, for example a sinusoidal steering input, the kinematic equations integrating IMU measurements work really well. These two factors combined make the observer without camera perform best. The camera signal is less accurate, which results in a drop in performance when adding a camera to the system under normal driving conditions.

When the model parameters are not correctly identified, the observer with CVA and DCE outperforms the observer without camera clearly. This was tested by running the observer with CVA and DCE versus the observer without camera on purposely wrongly identified tire parameters. When the tire force was scaled down by 66%, the Root Mean Square Error (RMSE) of the lateral velocity estimate for the observer with camera experienced an increase of 66%, while the RMSE of the observer without camera increased by 900%. These percentages are an average of four simulations, two times for dynamic driving conditions and two times for quasi-steady state driving conditions.

The observer with camera also performs better during drifting. By physically reducing rear wheel grip and letting the vehicle spin, it was shown that the observer with camera holds up much better in these conditions. The observer with camera is able to track sideslip angles of 90° and larger, where the observer without camera loses it. The RMSE almost doubles when removing the camera from the system for longitudinal as well as lateral velocity estimation.

As a result of the experiments, it is concluded that adding a camera to a vehicle velocity observer clearly has benefits if the goal is to create an observer that can deliver estimates of longitudinal velocity, lateral velocity and sideslip angle under all circumstances. When vehicle parameters are identified correctly, the observer with camera is not far behind the observer without camera in terms of performance. When there is a large error in the estimated tire parameters the observer with camera performs much better than the observer with out camera. This can be very helpful for observation in situations where online tire friction estimation is not quick enough, for exmaple when the road grip suddenly changes.

The used model had adequate assumptions that proved to be the right balance between modelling dynamics and not over-complicating the tuning and testing process. It was evaluated that the roll observer does a sizable contribution to improving observer performance and that the estimated vehicle parameters were accurate enough to enable estimation.

## 7-2 Recommendations

The experiments presented in this thesis indicate that it is worthwhile to further investigate the performance of the observation methodology. However, it can be stated that further testing of observer performance on this scaled testing platform is starting to reach its limits. Although the algorithm has not been implemented online, a representative test has been done where the maximum potential performance of the algorithm has been shown. If the final goal of the algorithm is for it to be used in consumer vehicles for autonomous drift control in ten years, then there are two areas where further research could have major impact.

The first is further developing the CVA to operate online in real time and to be able to handle representative driving conditions. Currently the speed of the vehicle was limited to $2.5\,\mathrm{m\,s^{-1}}$ in experiments and only the ground plane was tracked. Next steps could include the detection of moving obstacles, separating them from the ground plane or deliberately tracking the moving obstacles. Also online detection of the ground plane could be an interesting addition to remove the need for pitch and roll correction. Although the speed of the vehicle is low, lateral acceleration and yaw rate reached high values during experiments. It would be interesting to see how the proposed methodology holds up when vehicle speed becomes much larger, but lateral acceleration and yaw rate decrease. This should represent more common full scale driving conditions. The most efficient way to simulate these conditions would be to experiment on a larger sized vehicle.

The other area that would be interesting to test is how useful the output signal of the developed observer is for an autonomous drift controller. It is still unclear if the accuracy of the estimate coming from the developed controller is enough to enable autonomous drifting maneuvers. This can be tested by combining the developed observer with a controller. This can be done by either implementing it online on the BARC or by using the identification data from this report and use it in a (high-fidelity) simulation.

A final interesting step would be to test the correlation between small scale experiments on the BARC and full scale experiments on consumer vehicles. If comparable research would ever be done on consumer vehicles, the results of this project could be put into more perspective and its relevance can be stated for further research.

# Appendix A

# BARC information

In this appendix some information on the hardware and software of the used test platform can be found. The used test platform is the Berkeley Autonomous Race Car (BARC) and was developed at University of California, Berkeley. Most information on the BARC can be found on their GitHub. In the hardware section the used materials and links to their respective manufacturers website can be found. In the software section the programming of the vehicle and the used configuration of Robot Operating System (ROS) can be found. This chapter is not a complete description and manual how to setup a BARC, but merely a description of the most important elements of the BARC for this project.

## A-1 Hardware information

The main components of the BARC have been described from a estimation point of view in Chapter 2. The placement of the sensors on the vehicle can be found in Figure A-1. The corresponding component for each number can be found in Table A-1. Finally some identified or measured parameters of the BARC can be found in Table A-2.

**Table A-1:** BARC system parameters

| # | Component | Function | Type | Link |
|---|---|---|---|---|
| 1 | Main control unit | High level control, determine vehicle maneuvers, user interface | Hardkernel Odroid XU4 | Webshop |
| 2 | IMU | Measure planar acceleration, roll velocity and magnetic field strength | Hardkernel myAHRS+ | Webshop |
| 3 | Secondary control unit | Serial communication between Odroid and sensors | Arduino Nano (ATMega 328) | Webshop |
| 4 | Camera | Record images of ground plane | ELP US-BFHD01M | Manufacturer |
| 5 | Brushless DC Motor | Produce rear wheel torque | Unknown, part of Basher RZ-4 chassis | Webshop |
| 6 | Motor Control Unit | Control speed of motor | Unknown, part of Basher RZ-4 chassis | Webshop |
| 7 | Steering servo | Steer front wheels | E6001 | Webshop |
| 8 | Hall sensor | Measure wheel speed by detecting passing magnets | US1881 | Webshop |

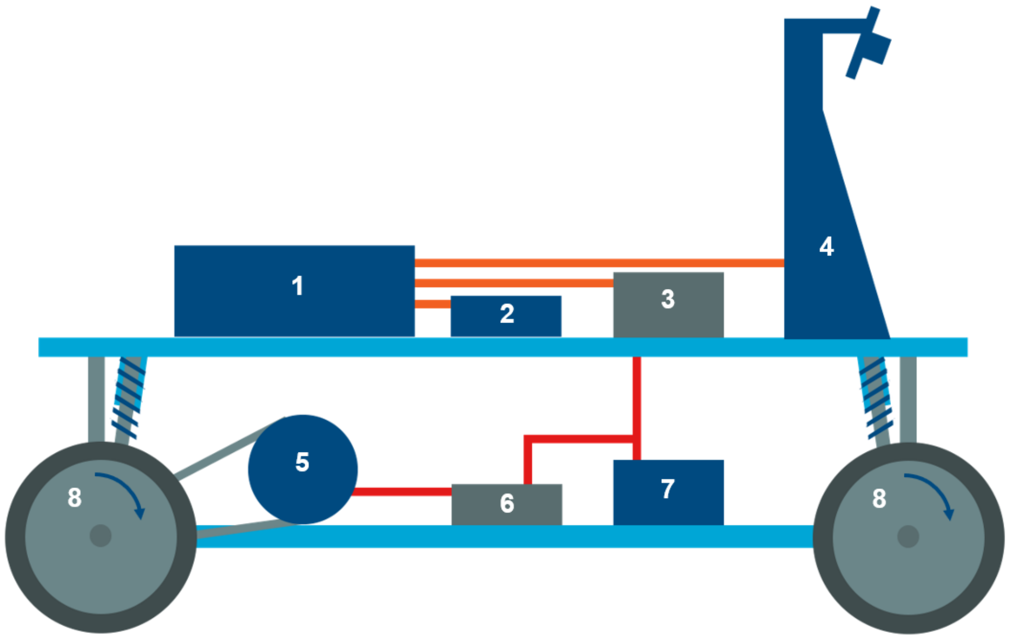


**Figure A-1:** Side view of the BARC. Legend of all labelled components can be found in Table A-1

**Table A-2:** BARC system parameters

| Symbol | Description | Value | Unit |
|---|---|---|---|
| $C_{\alpha,f}$ | Front cornering stiffness | 74.04 | $\mathrm{N\,rad^{-1}}$ |
| $C_{\alpha,r}$ | Rear cornering stiffness | 57.06 | $\mathrm{N\,rad^{-1}}$ |
| $l_f$ | Distance from centre of gravity to front axle | 0.145 | m |
| $l_r$ | Distance from centre of gravity to rear axle | 0.120 | m |
| $L$ | Total wheel base | 0.265 | m |
| $m$ | Vehicle mass | 1.980 | kg |
| $r_{\text{eff}}$ | Effective wheel radius | 0.034 | m |

## A-2 Software information

In this section the setup of the software running on the BARC is shown. At the core of the BARC lies the Odroid XU4. It runs Lubuntu 14.04, which is a lightweight version of Ubuntu. Within it, Robot Operating System (ROS) is used. ROS is a versatile tool for creating operating systems for robots. It allows for several nodes running in parallel to each other. Each node can publish messages on a topic and read messages from other topics. In Figure A-2 it can be seen that within ROS Indigo a variety of nodes is running. The Odroid XU4 is communicating with the Inertial Measurement Unit (IMU) and Camera over USB. The other sensors and actuators are controlled via an Arduino Nano. This is easier because the Arduino has analog interface pins and allows for serial communication, where the Odroid is more a small computer. Finally the Odroid XU4 can be accessed over WiFi with either SSH protocol or NoMachine (3rd party application).

Within ROS several nodes are running and communicating in parallel. In this way several agents can be created that are all tasked with their own subtasks. This allows for flexibility and easier development. The used nodes for the open loop experiments are shown in Figure A-3. A square represents a node and an oval a topic, which is a place where the messages of the nodes are posted. The nodes in Figure A-3 have the following function:

- High-level controller: set the desired rear wheel torque and steering angle based on a maneuver selected by the user. Post commands to `/ecu_cmd`.
- Low-level controller: convert the desired torque and steering angle posted on `/ecu_cmd` to a Pulse-Width-Modulation (PWM) signal that the actuators can handle. Conversion is done based on the identified relationships from Chapter 5. Post PWM commands to `/ecu_pwm`.
- Serial communication: Take the command from `/ecu_cmd` and deliver it to the actuators. Retrieve the amount of magnets passed at the wheel speed sensor and the time between the most recent passing of a magnet and post to respectively `/encoder` and `vel_est`.
- IMU readout: handle USB communication with IMU and post the measurements to `/imu/data_raw`.
- Camera recording: grab images from the camera and save them as `.png` image files on the flash storage. Save the time between the two most recent frames on the topic

`/frame_time` so that duplicate frames can be detected.

- Rosbag record: grab data from all available topics and save them in a `.rosbag` file on the flash storage. This file can then be taken to Matlab and used for simulation, together with the recorded images.

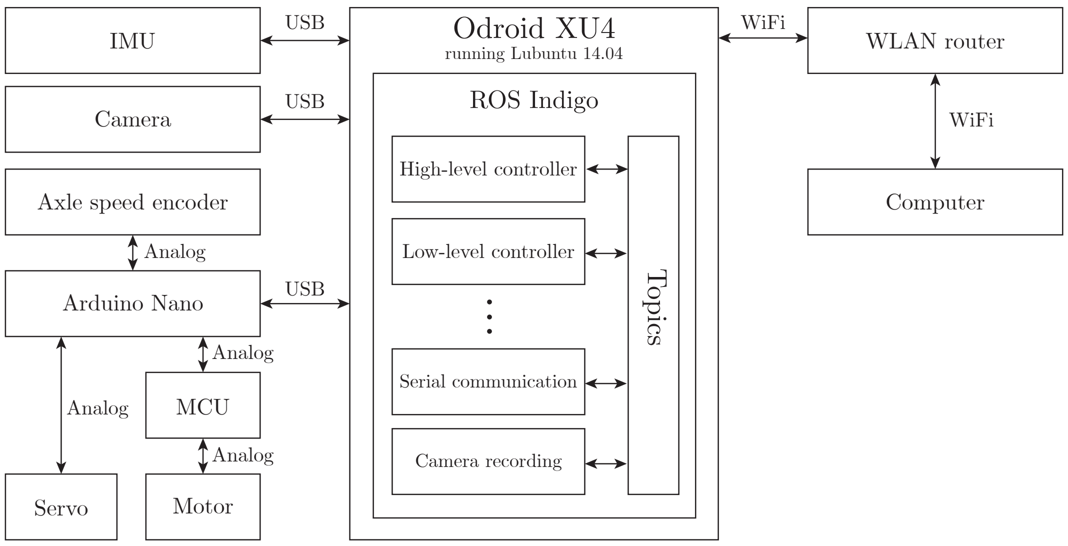


**Figure A-2:** Block diagram of all the components of the BARC and the way they are linked up. The software overview of the Odroid XU4 is a simplified version. During operation more nodes are present.

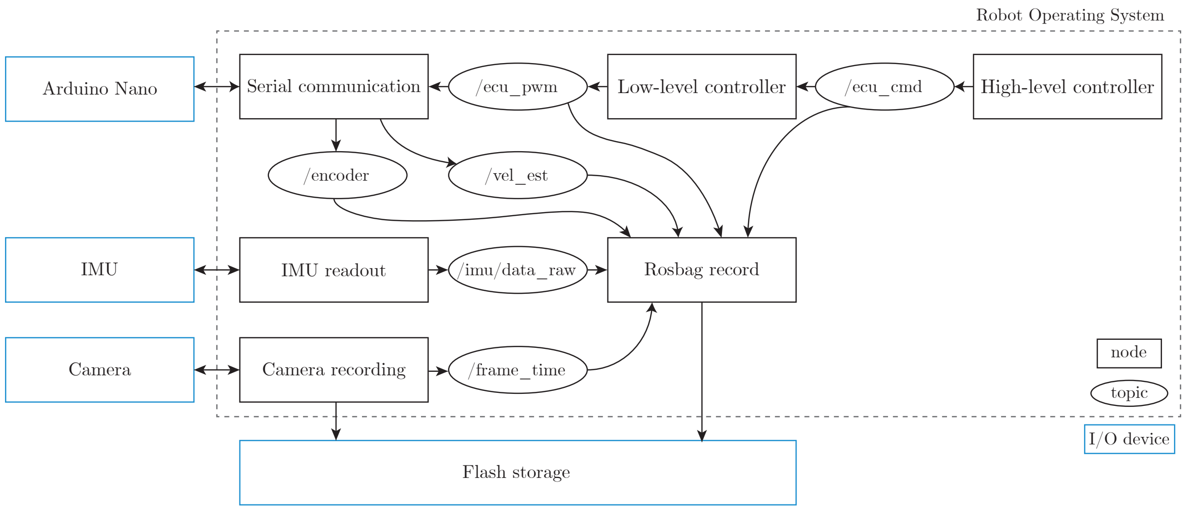


**Figure A-3:** Block diagram of all the topics and nodes running in ROS and the way they are linked with the real world

# Appendix B

# Mathematics

## B-1 Jacobian of output matrix

. The output vector for the hybrid dynamic-kinematic model was defined as:

$$h(\mathbf{x}) = \begin{bmatrix} V_\delta \\ a_y \\ V_x \\ \vdots \\ V_x \\ V_y \\ \vdots \\ V_y \end{bmatrix} = \begin{bmatrix} \sqrt{(V_x^2 + (V_{yf} - l_f\omega_z)^2)}\cos\left(\delta_f - \tan^{-1}\left(\frac{V_{yf}}{V_x}\right)\right) \\ \frac{1}{m}\left(F_{yf}\cos\delta_f + F_{yr}\right) \\ V_x \\ \vdots \\ V_x \\ V_y \\ \vdots \\ V_y \end{bmatrix} \tag{B-1}$$

With subfunctions $V_{yf}$ and $F_{yi}$ defined as:

$$V_{yf} = V_y - l_f\omega_z \tag{B-2}$$

$$F_{yi} = \frac{C_i}{k_i}\tanh k_i\alpha_i \quad i = \{f, r\} \tag{B-3}$$

$$\alpha_f = \left(\delta_f - \frac{V_y + l_f\omega_z}{V_x}\right) \quad \alpha_r = \left(-\frac{V_y - l_r\omega_z}{V_x}\right) \tag{B-4}$$

The Jacobian of the output vector can then be written as:

$$H = \begin{bmatrix} \frac{\partial h_{V_\delta}}{\partial V_x} & \frac{\partial h_{V_\delta}}{\partial V_y} \\ \frac{\partial h_{a_y}}{\partial V_x} & \frac{\partial h_{a_y}}{\partial V_y} \\ \frac{\partial h_{V_x}}{\partial V_x} & \frac{\partial h_{V_x}}{\partial V_y} \\ \vdots & \vdots \\ \frac{\partial h_{V_x}}{\partial V_x} & \frac{\partial h_{V_x}}{\partial V_y} \\ \frac{\partial h_{V_y}}{\partial V_x} & \frac{\partial h_{V_y}}{\partial V_y} \\ \vdots & \vdots \\ \frac{\partial h_{V_y}}{\partial V_x} & \frac{\partial h_{V_y}}{\partial V_y} \end{bmatrix} \tag{B-5}$$

$$\frac{\partial h_{V_\delta}}{\partial V_x} = \frac{V_x \cos(\delta - \arctan \frac{V_y - l_f\omega_z}{V_x})}{\sqrt{(V_y - l_f\omega_z)^2 + V_x^2}} - \frac{\sin(\delta - \arctan \frac{V_y - l_f\omega_z}{V_x})\sqrt{(V_y - l_f\omega_z)^2 + V_x^2}(V_y - l_f\omega_z)}{V_x^2 \left(\frac{(V_y - l_f\omega_z)^2}{V_x^2} + 1\right)} \tag{B-6}$$

$$\frac{\partial h_{V_\delta}}{\partial V_y} = \frac{\cos(\delta - \arctan \frac{V_y - l_f\omega_z}{V_x})(2V_y - 2l_f\omega_z)}{2\sqrt{(V_y - l_f\omega_z)^2 + V_x^2}} + \frac{\sin(\delta - \arctan \frac{V_y - l_f\omega_z}{V_x})\sqrt{(V_y - l_f\omega_z)^2 + V_x^2}}{V_x^2 \left(\frac{(V_y - l_f\omega_z)^2}{V_x^2} + 1\right)} \tag{B-7}$$

$$\frac{\partial h_{a_y}}{\partial V_x} = -\frac{1}{mV_x^2}\Big(C_f(\tanh^2(k_f\alpha_f) - 1)(V_y + l_f\omega_z) + C_r(\tanh^2(k_r\alpha_r) - 1)(V_y - l_r\omega_z)\Big) \tag{B-8}$$

$$\frac{\partial h_{a_y}}{\partial V_y} = \frac{1}{mV_x}\Big(C_f(\tanh^2(k_f\alpha_f) - 1) + C_r(\tanh^2(k_r\alpha_r) - 1)\Big) \tag{B-9}$$

$$\frac{\partial h_{V_x}}{\partial V_x} = 1 \qquad \frac{\partial h_{V_x}}{\partial V_y} = 0 \qquad \frac{\partial h_{V_y}}{\partial V_x} = 0 \qquad \frac{\partial h_{V_y}}{\partial V_y} = 1 \tag{B-10}$$

## B-2 Extrinsic matrix decomposition

In Section 3-1 the homography matrix $\mathcal{H}$ was introduced, that converted world points to image points by assuming zero height $Z = 0$ . It could be decomposed into a intrinsic parameter matrix $\mathcal{K}$ and a partial base extrinsic parameter matrix $\mathcal{E}'_0$:

$$R_0 = R_x(\phi_0)\ R_y(\theta_0)\ R_z(\psi_0) \tag{B-11}$$

$$\mathcal{H}_0 = \underbrace{\begin{bmatrix} f_x & \alpha & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathcal{K}} \underbrace{\begin{bmatrix} | & | & | \\ r_{1,0} & r_{2,0} & t_0 \\ | & | & | \end{bmatrix}}_{\mathcal{E}'_0} \tag{B-12}$$

The full extrinsic parameter matrix $\mathcal{E}_0$ for the resting vehicle can be retrieved by normalizing the first two columns of $\mathcal{E}'_0$ and finding a third column using the cross product. The angles of the camera pose can then be retrieved by using normal right hand rotation matrices, the used rotation sequence and some clever mathematics taken from [62]:

$$E = K^{-1}H = \begin{bmatrix} | & | & | \\ r_1 & r_2 & t \\ | & | & | \end{bmatrix} \tag{B-13}$$

$$\hat{r}_i = \frac{r_i}{|r_i|} \tag{B-14}$$

$$R = \begin{bmatrix} | & | & | \\ \hat{r}_1 & \hat{r}_2 & \hat{r}_1 \times \hat{r}_2 \\ | & | & | \end{bmatrix} \tag{B-15}$$

$$R = R_x(\phi)R_y(\theta)R_z(\psi) = \begin{bmatrix} c_\phi c_\theta & -c_\theta s_\psi & s_\theta \\ c_\phi s_\psi - c_\psi s_\phi s_\theta & c_\phi c_\psi + s_\phi s_\psi s_\theta & -c_\theta s_\phi \\ -s_\phi s_\psi - c_\phi c_\psi s_\theta & c_\phi s_\psi s_\theta - c_\psi s_\phi & c_\phi c_\theta \end{bmatrix} \tag{B-16}$$

$$\theta_0 = \operatorname{atan2}(r_{13}, \sqrt{r_{23}^2 + r_{33}^2}) \tag{B-17}$$

$$\psi_0 = \operatorname{atan2}(-r_{12}/\cos\theta, r_{11}/\cos\theta) \tag{B-18}$$

$$\phi_0 = \operatorname{atan2}(-r_{23}/\cos\theta, r_{33}/\cos\theta) \tag{B-19}$$

$$\tag{B-20}$$

The elements $r_{ij}$ correspond to row $i$, column $j$ of the rotation matrix. The letter $s$ indicates a sine term, the letter $c$ a cosine term. The `atan2` function is a normal arctan function, but by taking two separate arguments and looking at the sign of both arguments can derive the angle in a range of $\langle -\pi, \pi \rangle$. The base pose of the camera is now known. The extra roll and pitch angle of the vehicle can be added to these angles during operation and the whole procedure can be run in reverse to obtain an updated extrinsic parameter matrix $\mathcal{E}_k$ and updated homography matrix $\mathcal{H}_k$.

## B-3 Measurement noise covariance derivation

$$\begin{aligned}
E[v(k)v(k)^T] =& E[\frac{\epsilon^2_{x,k-1}}{dt^2} + \frac{\epsilon^2_{x,k}}{dt^2} + \epsilon^2_{y,k-1}\epsilon^2_{\omega,k} + \epsilon^2_{\omega,k}p^2_{y,k-1} + \epsilon^2_{y,k-1}\omega^2_{z,k} + \cancel{2\epsilon^2_{y,k-1}\epsilon_{\omega,k}\omega_{z,k}} \\
& \cancel{-\frac{2\epsilon_{x,k-1}\epsilon_{x,k}}{dt^2} + 2\epsilon_{y,k-1}\epsilon^2_{\omega,k}p_{y,k-1} + 2\epsilon_{y,k-1}\epsilon_{\omega,k}p_{y,k-1}\omega_{z,k}} \\
& \cancel{-\frac{2\epsilon_{x,k-1}\epsilon_{y,k-1}\epsilon_{\omega,k}}{dt} + \frac{2\epsilon_{y,k-1}\epsilon_{x,k}\epsilon_{\omega,k}}{dt} - \frac{2\epsilon_{x,k-1}\epsilon_{\omega,k}p_{y,k-1}}{dt}} \\
& \cancel{+\frac{2\epsilon_{x,k}\epsilon_{\omega,k}p_{y,k-1}}{dt} - \frac{2\epsilon_{x,k-1}\epsilon_{y,k-1}\omega_{z,k}}{dt} + \frac{2\epsilon_{y,k-1}\epsilon_{x,k}\omega_{z,k}}{dt}}]
\end{aligned} \tag{B-21}$$

$$E[v(k)v(k)^T] = E\left[\frac{\epsilon^2_{x,k-1}}{dt^2} + \frac{\epsilon^2_{x,k}}{dt^2} + \epsilon^2_{y,k-1}\epsilon^2_{\omega,k} + \epsilon^2_{\omega,k}p^2_{y,k-1} + \epsilon^2_{y,k-1}\omega^2_{z,k}\right] \tag{B-22}$$

$$= \frac{\sigma^2_{x,k-1} + \sigma^2_{x,k}}{dt^2} + \sigma^2_{y,k}\sigma^2_{\omega,k} + \sigma^2_{\omega,k}p^2_{y,k} + \sigma^2_{y,k}\omega^2_{z,k} \tag{B-23}$$

$$r_{i,x,k} = c_1\ \frac{\sigma^2_{x,k-1}}{dt^2} + c_2\ \frac{\sigma^2_{x,k}}{dt^2} + c_3\ \sigma^2_{y,k}\sigma^2_{\omega,k} + c_4\ \sigma^2_{\omega,k}p^2_{y,k} + c_5\ \sigma^2_{y,k}\omega^2_{z,k} \tag{B-24}$$

$$r_{i,x,k} = c_6\ \frac{\sigma^2_{y,k-1}}{dt^2} + c_7\ \frac{\sigma^2_{y,k}}{dt^2} + c_8\ \sigma^2_{x,k}\sigma^2_{\omega,k} + c_9\ \sigma^2_{\omega,k}p^2_{x,k} + c_{10}\ \sigma^2_{x,k}\omega^2_{z,k} \tag{B-25}$$

$$\sigma_{x,k-1} = f_{hom,x}(v_{k-1}) \tag{B-26}$$

$$\sigma_{x,k} = f_{hom,x}(v_k) + af_{detect}(q_{p,k-1}) + (1-a)f_{track}(\epsilon_{\text{track},k}) \tag{B-27}$$

$$\sigma_{y,k-1} = f_{hom,y}(v_{k-1}) \tag{B-28}$$

$$\sigma_{y,k} = f_{hom,y}(v_k) + af_{detect}(q_{p,k-1}) + (1-a)f_{track}(\epsilon_{\text{track},k}) \tag{B-29}$$

$$\tag{B-30}$$

# Bibliography

[1] T. Toroyan, “Global status report on road safety,” World Health Organization, Tech. Rep., 2015.

[2] European Commission, “European Commission, Annual Accident Report,” European Commission, Directorate General for Transport, Tech. Rep., 2017.

[3] L. Aarts, J. Commandeur, R. Welsh, S. Niesen, M. Lerner, P. Thomas, N. Bos, and R. Davidse, “Study on Serious Road Traffic Injuries in the EU,” Tech. Rep., 2016.

[4] European Commission, “Advanced driver assistance systems,” European Commission, Directorate General for Transport, Tech. Rep., 2016.

[5] J. N. Dang, “Statistical Analysis of the Effectiveness of Electronic Stability Control (Esc) Systems,” National Highway Traffic Safety Administration, Tech. Rep., 2007.

[6] M. Acosta, S. Kanarachos, and M. Blundell, “Vehicle agile maneuvering: From rally drivers to a finite state machine approach,” *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016*, 2017.

[7] D. Chindamo, B. Lenzo, and M. Gadola, “On the Vehicle Sideslip Angle Estimation: A Literature Review of Methods, Models, and Innovations,” *Applied Sciences*, vol. 8, no. 3, p. 355, 2018.

[8] “SAE Vehicle Dynamics Terminology J670,” pp. 1–73, 2008.

[9] H. B. Pacejka, *Tire and Vehicle Dynamics*, 2nd ed. SAE International, 2005.

[10] W. F. Milliken and D. L. Milliken, *Race Car Vehicle Dynamics*. SAE International, 1995.

[11] O. Galluppi, M. Corno, and S. M. Savaresi, “Mixed-kinematic body sideslip angle estimator for high performance cars,” in *European Control Conference*, no. 1, 2018, pp. 1–6.

[12] M. Abdulrahim, “On the Dynamics of Automobile Drifting,” *SAE Technical Paper*, no. 2006-01-1019, 2006.

[13] E. Velenis, E. Frazzoli, and P. Tsiotras, “On steady-state cornering equilibria for wheeled vehicles with drift,” *Proceedings of the IEEE Conference on Decision and Control*, pp. 3545–3550, 2009.

[14] C. B. Kuyt, *Sensor Fusion of Computer Vision and Stock Sensors for Vehicle Dynamic Sideslip Estimation (MSc Thesis).* Delft University of Technology, 2017.

[15] E. Velenis, P. Tsiotras, and J. Lu, “Optimality Properties and Driver Input Parameterization for Trail-braking Cornering,” *European Journal of Control*, vol. 14, no. 4, pp. 308–320, 2008.

[16] C. Voser, R. Y. Hindiyeh, and J. C. Gerdes, “Analysis and control of high sideslip manoeuvres,” *Vehicle System Dynamics*, vol. 48, no. SUPPL. 1, pp. 317–336, 2010.

[17] R. Y. Hindiyeh and J. Christian Gerdes, “A Controller Framework for Autonomous Drifting: Design, Stability, and Experimental Validation,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 136, no. 5, p. 051015, 2014.

[18] M. Acosta, S. Kanarachos, and M. E. Fitzpatrick, “On full MAGV lateral dynamics exploitation: Autonomous drift control,” in *IEEE 15th International Workshop on Advanced Motion Control*, 2018, pp. 529–534.

[19] F. Zhang, J. Gonzales, K. Li, and F. Borrelli, “Autonomous Drift Cornering with Mixed Open-loop and Closed-loop Control,” *IFAC Proceedings*, vol. 50, no. 1, pp. 1916–1922, 2017.

[20] K. Kritayakirana and J. C. Gerdes, “Autonomous vehicle control at the limits of handling,” *Int. J. Vehicle Autonomous Systems*, vol. 10, no. 4, pp. 271–296, 2012.

[21] J. Y. Goh and J. C. Gerdes, “Simultaneous stabilization and tracking of basic automobile drifting trajectories,” *IEEE Intelligent Vehicles Symposium, Proceedings*, vol. 2016-Augus, no. Iv, pp. 597–602, 2016.

[22] M. Gadola, D. Chindamo, M. Romano, and F. Padula, “Development and validation of a Kalman filter-based model for vehicle slip angle estimation,” *Vehicle System Dynamics*, vol. 52, no. 1, pp. 68–84, 2014.

[23] S. Antonov, A. Fehn, and A. Kugi, “Unscented Kalman filter for vehicle state estimation,” *Vehicle System Dynamics*, vol. 49, no. 9, pp. 1497–1520, 2011.

[24] L. H. Zhao, Z. Y. Liu, and H. Chen, “Design of a nonlinear observer for vehicle velocity estimation and experiments,” *IEEE Transactions on Control Systems Technology*, vol. 19, no. 3, pp. 664–672, 2011.

[25] H. Ren, S. Chen, G. Liu, and K. Zheng, “Vehicle state information estimation with the unscented Kalman filter,” *Advances in Mechanical Engineering*, vol. 2014, 2014.

[26] L. R. Ray, “Nonlinear State and Tire Force Estimation for Advanced Vehicle Control,” *IEEE Transactions on Control Systems Technology*, vol. 3, no. 1, pp. 117–124, 1995.

[27] P. Dixon, M. Best, and T. Gordon, "An Extended Adaptive Kalman Filter for Real-time State Estimation of Vehicle Handling Dynamics," *Vehicle System Dynamics*, vol. 34, no. 1, pp. 57–75, 2000.

[28] D. Piyabongkarn, R. Rajamani, J. A. Grogg, and J. Y. Lew, "Development and experimental evaluation of a slip angle estimator for vehicle stability control," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 1, pp. 78–88, 2009.

[29] H. F. Grip, L. Imsland, T. A. Johansen, T. I. Fossen, J. C. Kalkkuhl, and A. Suissa, "Nonlinear vehicle side-slip estimation with friction adaptation," *Automatica*, vol. 44, no. 3, pp. 611–622, 2008.

[30] J. Bechtloff, L. Koenig, and R. Isermann, "Cornering Stiffness and Sideslip Angle Estimation for Integrated Vehicle Dynamics Control," in *8th IFAC Symposium on Advances in Automotive Control AAC 2016*, 2016, pp. 297–304.

[31] F. Naets, S. van Aalst, B. Boulkroune, N. El Ghouti, and W. Desmet, "Design and Experimental Validation of a Stable Two Stage Estimator for Automotive Sideslip Angle and Tire Parameters," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 11, pp. 9727–9742, 2017.

[32] S. van Aalst, F. Naets., B. Boulkroune, W. D. Nijs, and W. Desmet, "An Adaptive Vehicle Sideslip Estimator for Reliable Estimation in Low and High Excitation Driving," *IFAC-PapersOnLine*, vol. 51, no. 9, pp. 243–248, 2018.

[33] J. Farrelly and P. Wellstead, "Estimation of vehicle lateral velocity," *Proceedings of the 1996 IEEE International Conference on Control Applications*, no. November, pp. 552–557, 1996.

[34] D. Selmanaj, M. Corno, G. Panzani, and S. M. Savaresi, "Vehicle sideslip estimation: A kinematic based approach," *Control Engineering Practice*, vol. 67, no. December 2016, pp. 1–12, 2017.

[35] A. K. Madhusudhanan, M. Corno, and E. Holweg, "Vehicle sideslip estimator using load sensing bearings," *Control Engineering Practice*, vol. 54, pp. 46–57, 2016.

[36] Y. Fukada, "Slip-Angle Estimation for Vehicle Stability Control," *Vehicle System Dynamics*, vol. 32, no. 4-5, pp. 375–388, 1999.

[37] A. Y. Ungoren, H. Peng, and H. E. Tseng, "A study on lateral speed estimation methods," *International Journal of Vehicle Autonomous Systems*, vol. 2, no. 1/2, pp. 126–144, 2004.

[38] F. Cheli, E. Sabbioni, M. Pesce, and S. Melzi, "A methodology for vehicle sideslip angle identification: comparison with experimental data," *Vehicle System Dynamics*, vol. 45, no. 6, pp. 549–563, 2007.

[39] W. Kruger, W. Enkelmann, and S. Rossle, "Real-Time Estimation and Tracking of Optical Flow," *Intelligent Vehicles '95 Symposium*, pp. 304–309, 1995.

[40] K. Yamaguchi, T. Kato, and Y. Ninomiya, "Vehicle Ego-Motion Estimation and Moving Object Detection using a Monocular Camera," *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 4, pp. 610–613, 2006.

[41] A. Jaegle, S. Phillips, and K. Daniilidis, “Fast, robust, continuous monocular egomotion computation,” *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2016-June, pp. 773–780, 2016.

[42] F. Harchut and B. Mueller-Bessler, “Camera based measurement of side slip angle in an automobile,” in *ASME Dynamic Systems and Control Conference*, 2009, pp. 1–6.

[43] T. Botha, P. Els, B. Jacobson, and A. Albinsson, “Vehicle motion measurements using front facing camera and digital image correlation,” in *Proceedings of the ASME Design Engineering Technical Conference*, vol. 3, Charlotte, North Carolina, 2016.

[44] Y. Wang, B. M. Nguyen, H. Fujimoto, and Y. Hori, “Vision based Multi-rate Estimation and Control of Body Slip Angle for Electric Vehicles,” in *38th Annual Conference on IEEE Industrial Electronics Society*, 2012, pp. 4278–4283.

[45] Y. Wang, B. M. Nguyen, P. Kotchapansompote, H. Fujimoto, and Y. Hori, “Vision-based vehicle body slip angle estimation with multi-rate Kalman filter considering time delay,” *IEEE International Symposium on Industrial Electronics*, no. February, pp. 1506–1511, 2012.

[46] Y. Wang, B. M. Nguyen, H. Fujimoto, and Y. Hori, “Multirate estimation and control of body slip angle for electric vehicles based on onboard vision system,” *IEEE Transactions on Industrial Electronics*, vol. 61, no. 2, pp. 1133–1143, 2014.

[47] University of California - Berkely, “BARC project.” [Online]. Available: http://www.barc-project.com/

[48] A. Castro, S. Braga, and M. Neto, “A Method To Estimate Parameters of Longitudinal and Lateral Dynamics of Ground Vehicles,” *22nd International Congress of Mechanical Engineering*, no. August 2017, pp. 6324–6336, 2013.

[49] “OpenCV (Open Source Computer Vision Library).” [Online]. Available: https://opencv.org/

[50] Z. Zhang, “A Flexible New Technique for Camera Calibration (Technical Report),” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2002.

[51] C. Ricolfe-Viala and A.-J. Sanchez-Salmeron, “Lens distortion models evaluation,” *Applied Optics*, vol. 49, no. 30, p. 5914, 2010.

[52] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. B. Zimmerman, and K. Zuiderveld, “Adaptive histogram equalization and its variations,” *Computer Vision, Graphics, and Image Processing*, vol. 39, no. 3, pp. 355–368, 9 1987.

[53] N. Nourani-Vatani, P. V. K. Borges, and J. M. Roberts, “A study of feature extraction algorithms for optical flow tracking,” *Australasian Conference on Robotics and Automation, ACRA*, pp. 3–5, 2012.

[54] J. Shi and C. Tomasi, “Good Features to Track,” in *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Seattle, 1994, pp. 593–600.

[55] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *7th International Joint Conference on Artificial Intelligence*, 1981, pp. 674–679.

[56] J.-y. Bouguet, V. Tarasenko, B. D. Lucas, and T. Kanade, "Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm," Intel Corporation, Tech. Rep., 2001.

[57] E. Yuan, "Coarse to Fine Optical Flow." [Online]. Available: http://eric-yuan.me/coarse-to-fine-optical-flow/

[58] Z. Kalal, K. Mikolajczyk, and J. Matas, "Forward-backward error: Automatic detection of tracking failures," in *2010 20th International Conference on Pattern Recognition.* IEEE, 8 2010, pp. 2756–2759.

[59] R. K. Mehra, "On the Identification of Variances and Adaptive Kalman Filtering," *IEEE Transactions on Automatic Control*, vol. 15, no. 2, pp. 175–184, 1970.

[60] B. Zheng, P. Fu, B. Li, and X. Yuan, "A robust adaptive unscented kalman filter for nonlinear estimation with uncertain noise covariance," *Sensors (Switzerland)*, vol. 18, no. 3, 2018.

[61] Optitrack, "Optitrack Prime 13W Camera." [Online]. Available: https://optitrack.com/products/prime-13w/

[62] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation.* CRC-Press, 1994, vol. 29.

[63] M. Brunner, "Repetitive Learning MPC and its application in autonomous race driving (MSc Thesis)," Univserity of California, Berkely, Tech. Rep., 2017. [Online]. Available: https://github.com/maxb91/Master-thesis

# Glossary

## List of Acronyms

| | |
|---|---|
| **ADAS** | Advanced Driver Assistance Systems |
| **BARC** | Berkeley Autonomous Race Car |
| **CLAHE** | Contrast Limited Adaptive Histogram Equalization |
| **CVA** | Computer Vision Algorithm |
| **DARE** | Discrete time Algebraic Ricatti Equation |
| **DCE** | Dynamic Covariance Estimation |
| **EKF** | Extended Kalman Filter |
| **IMU** | Inertial Measurement Unit |
| **IPLKOF** | Iterative Pyramidal Lucas Kanade Optical Flow |
| **LPV** | Linear Parameter Varying |
| **MCS** | Motion Capture System |
| **NRMSE** | Normalized Root Mean Square Error |
| **RMSE** | Root Mean Square Error |
| **PWM** | Pulse Width Modulation |
| **ROS** | Robot Operating System |
| **UKF** | Unscented Kalman Filter |
| **VAF** | Variance Accounted For |

## List of Symbols

### Greek Symbols

| | |
|---|---|
| $\alpha$ | Wheel slip angle |
| $\beta$ | Body sideslip angle |
| $\delta$ | Tire steering angle |
| $\omega$ | Rotational velocity |
| $\phi$ | Vehicle roll angle |
| $\psi$ | Vehicle yaw angle |
| $\sigma^2$ | Variance of a stochastic signal |
| $\Gamma_k$ | Discrete input matrix at sample $k$ |
| $\Phi_k$ | Discrete state transition matrix at sample $k$ |
| $\lambda$ | Longitudinal wheel slip ratio |

### Latin Symbols

| | |
|---|---|
| $\mathbf{u}_k$ | Input vector at sample $k$ |
| $\mathbf{v}_k$ | Measurement noise vector at sample $k$ |
| $\mathbf{w}_k$ | Process noise vector at sample $k$ |
| $\mathbf{x}_k$ | State vector at sample $k$ |
| $\mathbf{z}_k$ | Measurement signal vector |
| $\mathcal{E}$ | Extrinsic parameter matrix |
| $\mathcal{H}$ | Homography matrix |
| $\mathcal{K}$ | Intrinsic parameter matrix |
| $a$ | Acceleration |
| $A(t)$ | LPV state transition matrix at time $t$ |
| $B$ | Constant input matrix |
| $b_\phi$ | Rotational roll damping |
| $c_x, c_y$ | Location of optical center of a camera |
| $C_{\alpha,i}$ | Cornering stiffness for tire i |
| $d_{p_k}$ | Forward-backward projection error for point $p$ tracked $p_{k-1}$ to $p_k$ |
| $f_i$ | Focal distance along axis $i$ |
| $F_k$ | Camera frame number $k$ |
| $F_x$ | Longitudinal tire force |
| $F_y$ | Lateral tire force |
| $g$ | Gravitational acceleration, $g = 9.81\,\mathrm{m\,s^{-2}}$ |
| $h(\mathbf{x})$ | Nonlinear output function based on state vector $\mathbf{x}$ |
| $H_k(\mathbf{x})$ | Jacobian of nonlinear output function $h(\mathbf{x})$ at sample $k$ |
| $h_s$ | Distance from centre of mass to roll centre |
| $I_{xx}$ | Moment of inertia of a 3 dimensional system rotating around $x$ axis |
| $K_\phi$ | Rotational roll stiffness |

| | |
|---|---|
| $K_k$ | Kalman Filter gain at sample $k$ |
| $K_{US}$ | Understeer gradient |
| $l_i$ | Distance from centre of gravity to axle $i$ |
| $m$ | Vehicle mass |
| $m_s$ | Sprung mass of the vehicle |
| $N_p$ | Number of tracked points per frame |
| $p(u, v)$ | Pixel location $p$ |
| $P(X, Y, Z)$ | World point location $P$ |
| $P_k$ | State covariance estimate at sample $k$ |
| $Q_k$ | Process noise covariance matrix at sample $k$ |
| $q_{p_{k-1}}$ | Corner quality for point $p_{k-1}$ tracked from frame $k-1$ to $k$ |
| $R_k$ | Measurement noise covariance matrix at sample $k$ |
| $T_s$ | Sample time |
| $u$ | Horizontal pixel position |
| $V$ | Velocity |
| $v$ | Vertical pixel position |

## Subscripts

| | |
|---|---|
| $1..i..N_p$ | Point number in an array of tracked points |
| $1..k..k_{\max}$ | Sample number in an array of samples |
| $f, r$ | Vehicle front or rear axle |
| $k\|k-1$ | Calculation made at sample $k$ with information of sample $k-1$ |
| $X, Y, Z$ | Global reference system axes |
| $x, y, z$ | Vehicle reference system axis, respectively longitudinal, lateral and vertical axis. |

## Superscripts

| | |
|---|---|
| $\bar{x}$ | Quantity $x$ that is scaled using a heuristic function |
| $\dot{x}$ | Time derivative of quantity $x$ |
| $\hat{x}$ | Estimate of quantity $x$ |
| $\tilde{x}$ | Indication that signal $x$ contains an error |

# Index