ROYAL INSTITUTE
OF TECHNOLOGY

Lab 2 tutorial
Lateral Dynamics – State estimation

SD2231 – Applied vehicle dynamics control

March 17, 2016

Mikael Nybacka
Mats Jonasson

Postal address
Royal Institute of Technology
KTH Vehicle Dynamics
SE-100 44 Stockholm
Sweden

Visiting address
Teknikringen 8
Stockholm

Telephone
+46 8 790 6000
Telefax
+46 8 790 9304

Internet
www.ave.kth.se

Contents

1 Introduction

In todays vehicles there is a lot of functions that need information about the vehicles current state, especially the states longitudinal and lateral vehicle velocity. These are used in the control of vehicle functions such and Antilock braking system (ABS), Electronic Stability Control (ESC), Roll-over protection, Torque Vectoring, etc.

For the ESC and the Torque Vectoring it is of interest to have both the longitudinal velocity and the lateral in order to keep track of the vehicle sideslip. Either in order to create a very safe vehicle that do not allow any sideslip or a designed sideslip for a more "fun to drive" vehicle that allow vehicle sideslip "drifting" to let us say 15 deg for a very fun vehicle.

For the future there is a lot of discussions about vehicle platooning or even fully automated vehicles since it has the possibility to reduce environmental impact and increase traffic safety. However, such vehicles need to have full awareness of its own states and also surrounding environment.

For railway vehicles one example of the use of observers are when there is a need to build up a model that is able to estimate if there is a failure in a component or subsystem. In those cases a model of the subsystem is created and used in an observer to estimate the true output of that subsystem, when there is a sudden discrepancy or large error the fault or failure can be detected.

So we hope that this course and this exercise will give you a bit more knowledge on how to use your vehicle dynamics knowledge and apply that in different areas of vehicle control, specifically how to build observers and learn more about how they work and how they are built up.

This laboratory assignment gives you knowledge in vehicle state estimation in the context of vehicle motion control. The laboratory intends to give insight methods to estimate motion variables, building observers. In this lab, we will as an example, focus on one important vehicle state mentioned namely vehicle side-slip angle. This state is important since it gives information about how much the vehicle skids, and hence, this state forms a metric that is used to control lateral vehicle stability. Side-slip can be measured, but for, as an example, series production cars it is deemed to be too expensive. Instead side-slip is estimated by using other information, typically data from an inertial measurement unit (IMU), which provides measured quantities about the car body's accelerations and angular velocities.

2 Assignment

2.1 Theory

2.1.1 Body side-slip

In vehicle dynamics, it is convenient to use a coordinate system that follows the vehicle when it is turning. This coordinate system is here called "vehicle fixed coordinate system", which differs from the "global coordinate system", which is fixed relative to the earth. In the fixed coordinate system, the vehicle's centre of gravity (COG) moves longitudinally with a velocity v_x and laterally with a velocity v_y, which is illustrated in Figure 1. The drift sideways caused by the lateral velocity is called body side-slip and is defined as

$$\beta = \arctan\left(\frac{v_y}{v_x}\right) \tag{1}$$

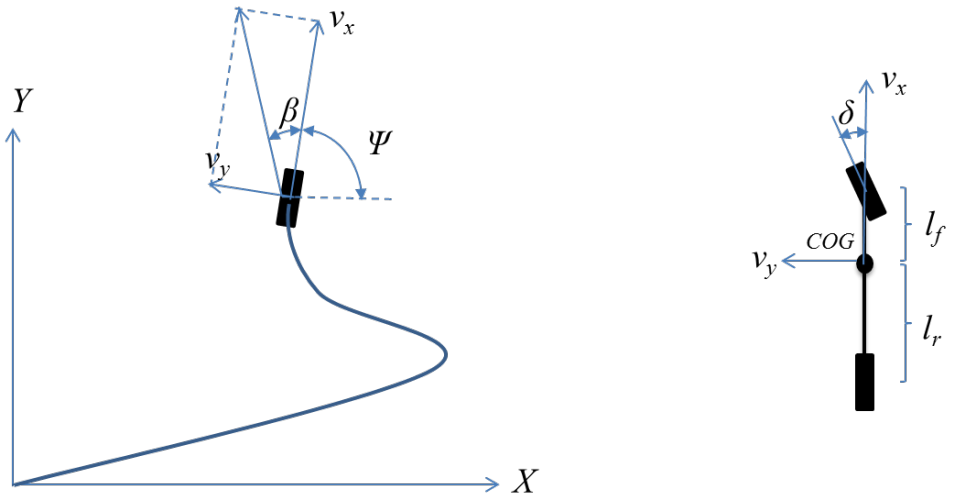

Figure 1. Side-slip angle β in centre of gravity.

The vehicle fixed coordinate system used in this lab follows the ISO definition, which means that x-axis is pointed forward, y-axis is pointed to the left and z-axis is pointed upwards.

2.1.2 Body side-slip from the bicycle model

This section describes how body side-slip can be physically modelled by a mathematical model referred to as the bicycle model. For details about the model, its derivation and limitations readers are referred to standard literature in the field and to the SD2225 course literature that is uploaded on the Bilda course webpage. The model, as it is formulated in Equation (2), describes the lateral motion in steady state for a tyre to road steering input δ. The degrees of freedom for this model consist of a yaw and a lateral motion component, with the corresponding state variables lateral velocity v_y and yawrate $\dot{\psi}_z$. The

longitudinal vehicle velocity v_x is assumed to be constant. The model is derived by assuming only one front wheel and one rear wheel and no centre of gravity height. The vehicle has a mass m and distances from centre of gravity to the front axle and rear axle are denoted l_f and l_r respectively. The tyres are modelled with a front and rear cornering stiffness C_{12} and C_{34} respectively.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \frac{\partial \dot{x}_1}{\partial x_1} & \frac{\partial \dot{x}_1}{\partial x_2} \\ \frac{\partial \dot{x}_2}{\partial x_1} & \frac{\partial \dot{x}_2}{\partial x_2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \frac{\partial \dot{x}_1}{\partial u} \\ \frac{\partial \dot{x}_2}{\partial u} \end{bmatrix} u$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -\frac{C_{34}-C_{12}}{mv_x} & -\frac{v_x^2 m - l_r C_{34} + l_f C_{12}}{mv_x} \\ -\frac{l_f C_{12} - l_r C_{34}}{I_z v_x} & -\frac{l_f^2 C_{12} + l_r^2 C_{34}}{I_z v_x} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \frac{C_{12}}{m} \\ \frac{l_f C_{12}}{I_z} \end{bmatrix} u \quad (2)$$

Where, $\dot{x}_1 = \dot{v}_y$ and $\dot{x}_2 = \ddot{\psi}_y$. Note that Equation (2) is not valid if there are other inputs, e.g. differential braking from an Electronic Stability Control system. For simplicity it is here assumed that there is a linear relationship between the steering-wheel angle *SWA* and the tyre to road steering angle such as

$$SWA = K_s \delta \quad (3)$$

Using cramer's rule to solve Equation (2) for x_1 and then combining with Equation (3) where $\dot{x}_1 = \dot{x}_2 = 0$, meaning the model is derived from steady state, gives a model-based expression of vehicle lateral velocity

$$v_y^{\text{mod}}(SWA, v_x) = \frac{v_x \left(l_r \left(l_f + l_r \right) C_{12} C_{34} - l_f C_{12} m v_x^2 \right)}{K_s \left(\left(l_f + l_r \right)^2 C_{12} C_{34} + m v_x^2 \left(l_r C_{34} - l_f C_{12} \right) \right)} \cdot SWA \quad (4)$$

Note there might be singularity in the denominator for certain values.
Commonly $v_y << v_x$, and hence small angle approximation can often be applied to Equation (1) without causing large error. Finally, the model-based body side-slip can be expressed as

$$\beta_y^{\text{mod}} \approx \frac{v_y^{\text{mod}}}{v_x} \quad (5)$$

Note the singularity for small v_x, which could results in unrealistic solutions when driving slowly.

2.1.3 Body side-slip by integrating lateral acceleration

Lateral velocity was in Section 2.1.2 estimated using the bicycle model, which possess a physical model of the vehicle. Another approach is to use kinematic relations and use measurements from the vehicle's inertial measurement unit (IMU), which measure the carbody's accelerations and angular speeds. The kinematics for a rigid body in lateral direction is governed by

$$a_y = \dot{v}_y + \dot{\psi}_z v_x - \dot{\varphi}_x v_z + g\sin(\varphi_x)\cos(\theta_y) \tag{6}$$

Where a_y is vehicle's lateral acceleration in COG. The velocities v_x, v_y, v_z are the vehicle's longitudinal, lateral and vertical velocities in COG. g is the gravity constant on earth and the carbody's Euler roll angle and pitch angle relative to the sea surface is denoted φ_x and θ_y respectively.

Moreover, the angular roll and yaw velocity are denoted $\dot{\varphi}_x$ and $\dot{\psi}_z$ respectively. Equation (6) describes the contributors to the lateral acceleration, which originates from the time derivative of lateral velocity plus cross products from angular velocities and transverse speeds. There is also a component from gravity, which will contaminate the lateral accelerometer reading during non-planar motion.

Assuming that vertical speed and the pitch Euler angle are both small, the lateral speed is approximated as

$$v_y^{kin} \approx \int_0^T \left(a_y - \dot{\psi}_z v_x - g\sin\left(\varphi_x\right)\right)dt \tag{7}$$

The name *kin* in v_y^{kin} indicates that the lateral velocity comes from a kinematic relation. Note that a_y and $\dot{\psi}_z$ are measured entities and that v_x here is assumed to come from another estimator in the vehicle's control architecture. The Euler roll angle stems from body roll due to suspension or/and road banking. Assuming that body roll from suspension increases linearly relative to lateral acceleration and that road banking is negligible, the lateral speed is found to be

$$v_y^{kin} \approx \int_0^T \left(a_y - \dot{\psi}_z v_x - g\sin\left(\tfrac{a_y}{g} K_{roll}\right)\right)dt \approx \int_0^T \left(a_y\left(1 - K_{roll}\right) - \dot{\psi}_z v_x\right)dt \tag{8}$$

Where K_{roll} is the vehicle's roll gradient in [rad/g] and is assumed that Euler roll angle is relatively small, this roll gradient needs to be multiplied with lateral acceleration in g's hence the a_y/g term. Note that the integral in Equation (7) is exposed to drift primarily due to the presence of lateral sensor offset; e.g. the sensor reading is not true. As a consequence, integration time must be held short.

2.1.4 Wash-out filter

The body side-slip estimated from the bicycle model is valid for steady state motion and estimation errors are expected for transient manoeuvres. The method where lateral acceleration is integrated is, on the other hand, suffering from drift. Having said that, it is possible to fuse information from both methods, this can be done using a wash-out filter where low and high pass information can be retrieved from the two methods according to

$$v_y = LP\left(v_y^{\text{mod}}\right) + HP\left(v_y^{kin}\right) \tag{9}$$

Where LP means low pass filtering of the lateral velocity from the bicycle model and HP means high pass filtering of the lateral velocity from the kinematic relation. Using the first order low and high pass filter with a time constant T according to the following transfer functions

$$H_{LP}(s) = \frac{1}{1+sT} \tag{10}$$

$$H_{HP}(s) = 1 - H_{LP}(s) = \frac{sT}{1+sT} \tag{11}$$

the lateral velocity is filtered as described in Equation (9)

$$v_y = \frac{1}{1+sT} v_y^{\text{mod}} + \frac{sT}{1+sT} v_y^{kin} = \frac{1}{1+sT}\left(v_y^{\text{mod}} + sTv_y^{kin}\right) \tag{12}$$

which together with the insertion of Equation (8) results in

$$v_y = \frac{1}{1+sT}\left(v_y^{\text{mod}} + T\left(a_y\left(1-K_{roll}\right) - \dot{\psi}_z v_x\right)\right) \tag{13}$$

As seen, the filtering in Equation (13) ends with a low pass filter. The filter coefficient T is a tunable parameter, which determines the balance between contributions from the bicycle model versus the measurement. T is the key for finding a good observer using this method but also a challenge, generally additional logic expressions need to be derived in order to build a good observer using this method.

2.1.5 Coordinate transformation

The equations in Sections above are all defined in COG and often it is not possible to mount IMU in this position. To transform lateral acceleration from sensor position to COG the following relation between them can be used, considering the COG in front of IMU:

$$\begin{aligned} a_y^{COG} &\approx a_y^{IMU} + \hat{\omega} \times \hat{r} \approx a_y^{IMU} + (0,0,\ddot{\psi}_z) \times (r_x, r_y, 0) \\ a_y^{COG} &\approx a_y^{IMU} + r_x \ddot{\psi}_z - r_y \ddot{\psi}_z \end{aligned} \tag{14}$$

Where $[r_x, r_y]$ are the longitudinal and lateral distances from IMU to COG position. Equation (14) is an approximation which is valid only when there is no roll or pitch motion and when the IMU is mounted without any misalignment. Note that angular speeds, such as yaw-rate, do not need to be adjusted due to another position of IMU than COG.

Lateral velocity needs also to be transformed when IMU position differs longitudinally from the one of COG. The relation between them reads

$$v_y^{COG} \approx v_y^{IMU} + r_x \dot{\psi}_z \tag{15}$$

2.1.6 Kalman filtering

Kalman filters have been around since the 1960's when Rudolf E. Kálmán published his paper on discrete-data linear filtering problem. Since that time there has been a lot of research and implementations of various Kalman filters for different applications. Some of the various filters originating from Kálmáns original work are listed below:

- Extended Kalman Filter
- Invariant extended Kalman Filter
- Unscented Kalman Filter
- Schmidt-Kalman Filter
- Ensemble Kalman Filter
- Hybrid Kalman Filter
- Kalman-Bucy filter
- And a lot more.

Some of the filters are more suitable for linear problems where the model of the system is linear and others are designed for models that are non-linear. Some filters of the ones listed above are also designed for continuous time where others are for discrete time.
So it is safe to say that we will not be able to give you a comprehensive overview of all filters and how they work. What we would like to provide with this exercise is a more practical knowledge on how to apply Kalman Filters for vehicle control applications as part of the observer in control systems.

2.1.6.1 Example of Kalman Filter implementation

So let us look at an example for a simple linear Kalman Filter (KF) where we would like to estimate the roll-angle of an Inertial Measuring Unit (IMU) that measures Accelerations in 3 axis (a_x, a_y and a_z) and Angular rates in 3 axis (roll-rate ($\dot{\varphi}$), pitch-rate ($\dot{\theta}$), yaw-rate ($\dot{\psi}$)).[1] [2]

Let us start by formulating the governing equations for roll angle and roll-rate bias:

$$f_1 : \varphi_k = \varphi_{k-1} - \dot{\varphi}_{b,k-1}\Delta t + \dot{\varphi}_k \Delta t + w_\varphi \tag{16}$$

$$f_2 : \dot{\varphi}_{b,k} = \dot{\varphi}_{b,k} + w_{\varphi_b} \tag{17}$$

So the new angle is the last angle minus the bias-angle plus the angle addition from the gyro reading. The bias rate is just equal to the last updated value since we cannot measure that. Seams reasonable right? So now we can build the state-space system.

The state of the system can be expressed with the following discrete functions:

$$x_k = A_k x_{k-1} + B_k u_k + w_k \tag{18}$$

$$y_k = C_k x_k + v_k \tag{19}$$

[1] http://blog.tkjelectronics.dk/2012/09/a-practical-approach-to-kalman-filter-and-how-to-implement-it/

[2] D. Simon, Kalman filtering, *Embedded Systems Programming*, pp. 72-79, June 2001.

where x_{k-1} is the state vector at time-step *k-1* that is given by:

$$x_{k-1} = \begin{bmatrix} \varphi \\ \dot{\varphi}_b \end{bmatrix}_{k-1} \tag{20}$$

and φ is the roll-angle and $\dot{\varphi}_b$ is the roll-rate bias that we need to estimate since the gyro signals will drift with time. Similarly the input u_k is simply just the roll-rate from the gyro $\dot{\varphi}$.

A, B and C is the state transition matrix, input matrix and measurement matrix respectively and is defined as:

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 1 & -\Delta t \\ 0 & 1 \end{bmatrix} \tag{21}$$

$$B = \begin{bmatrix} \frac{\partial f_1}{\partial u} \\ \frac{\partial f_2}{\partial u} \end{bmatrix} = \begin{bmatrix} \Delta t \\ 0 \end{bmatrix} \tag{22}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{23}$$

Equation (19) is called the output equation or measurement equation and here we map the state vector to the measurement. Here the measurement matrix Equation (19) is very simple since we will measure the angle directly using the accelerometer signals a_y and a_z, just $\arctan\left(a_y / a_z\right)$. So the measurement matrix does not have to include any Δt.

w_k and v_k is called the process noise and measurement or observation noise respectively and here we assume that the average value of w_k and v_k is zero and that w_k and v_k are independent random variables.

Then the process and measurement noise covariance matrices Q_k and R_k are defined as:

$$Q_k = E\left(w_k \cdot w_k^T\right) \tag{24}$$

$$R_k = E\left(v_k \cdot v_k^T\right) \tag{25}$$

where $E(\cdot)$ means the expected value, i.e. weighted average of all possible values, and ^{T} means transpose.

Now we take a look at the Kalman Filter equations. These can be divided in two steps, “predict” and “update”. The predict step uses state estimate from previous time step to produce an estimate of the state at current time step. This

state is also called the "a priori" state estimate because it does not include the observation information from the current timestep. In the update step the current a priori prediction is combined with current observation information to refine the state estimate. Then the estimate is called "a posteriori" state estimate.

Predict step:

Predict (a priori) state est.
$$\hat{x}_{k|k-1} = A_k \hat{x}_{k-1|k-1} + B_k u_k \quad (26)$$

Predict (a priori) est. cov.
$$P_{k|k-1} = A_k P_{k-1|k-1} A_k^T + Q_k \quad (27)$$

Update step:

Innovation or meas. residual
$$\tilde{y}_k = y_k - C_k \hat{x}_{k|k-1} \quad (28)$$

Innovation (or residual) cov.
$$S_k = C_k P_{k|k-1} C_k^T + R_k \quad (29)$$

Optimal Kalman Gain
$$K_k = P_{k|k-1} C_k^T S_k^{-1} \quad (30)$$

Updated (a posteriori) state est
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \tilde{y}_k \quad (31)$$

Updated (a posteriori) est. cov.
$$P_{k|k} = \left(I - K_k C_k\right) P_{k|k-1} \quad (32)$$

where $^{-1}$ means matrix inversion and ^{T} means transpose. The K_k is called the Kalman gain and $P_{k|k-1}$ is called the estimation error covariance.

The state estimate equation (26) is fairly intuitive. The first term used to derive the state estimate at time *k* is just A times the state estimate at time *k-1*, plus B times the known input at time *k-1*. This would be the state estimate if we did not have a measurement. In other words, the state estimate would propagate in time just like the state vector in the system model.

Inspection of equation (30) shows that if the measurement noise is large v_k, R_k will be large, meaning that K_k will be small and we would not give much credibility to the measurement $\tilde{y}_k$ when computing the next $\hat{x}_{k|k}$. On the other hand, if the measurement noise is small, R_k will be small, so K_k will be large and we will give a lot of credibility to the measurement when computing the next $\hat{x}_{k|k}$.

Lets then look at the implementation. Since we only have one measure $\arctan\left(a_y / a_z\right)$, and we assume that the standard deviation (*std*) will not change over time the noise covariance will simply be the variance of $\arctan\left(a_y / a_z\right)$:

$$R_k = E\left(v_k \cdot v_k^T\right) = std\left(u\right)^2 = std\left(\arctan\left(a_y / a_z\right)\right)^2 = 0.0021^2 \quad (33)$$

The process noise is a bit more difficult to calculate and usually with a larger system you will have to start with a value and then tune the process noise so that it will give better results, this can either be done by using trial and error

approach or with the help of system identification like Autocovariance Least-Squares[3].

For this example we assume that the angle and the bias is independent from each other so any product of the bias and angle noise will be set to zero. We also measure the roll-rate noise to be 0.0056, but the roll-rate bias we do not know so we choose a value that will give good results which in our case is 0.03. From the state space equations we also see that roll-angle is proportional to Δt times the roll-rate, where roll-rate noise is 0.0056. So if writing out the process noise covariance it will be the following with a time step Δt of 0.01:

$$Q_k = E\left(w_k \cdot w_k^T\right) = E\left(\begin{bmatrix} w_\varphi \\ w_{\dot{\varphi}_b} \end{bmatrix}\begin{bmatrix} w_\varphi & w_{\dot{\varphi}_b} \end{bmatrix}\right) = \tag{34}$$

$$E\begin{bmatrix} w_\varphi^2 & w_\varphi w_{\dot{\varphi}_b} \\ w_\varphi w_{\dot{\varphi}_b} & w_{\dot{\varphi}_b}^2 \end{bmatrix} = \begin{bmatrix} (0.0056)^2 \cdot (0.01)^2 & 0 \\ 0 & (0.03)^2 \end{bmatrix}$$

The rest is just implementation in Matlab and the implementation of a Kalman Filter in Matlab can be seen in referenced material[4], see Figure 2 for results from VBOX IMU data.

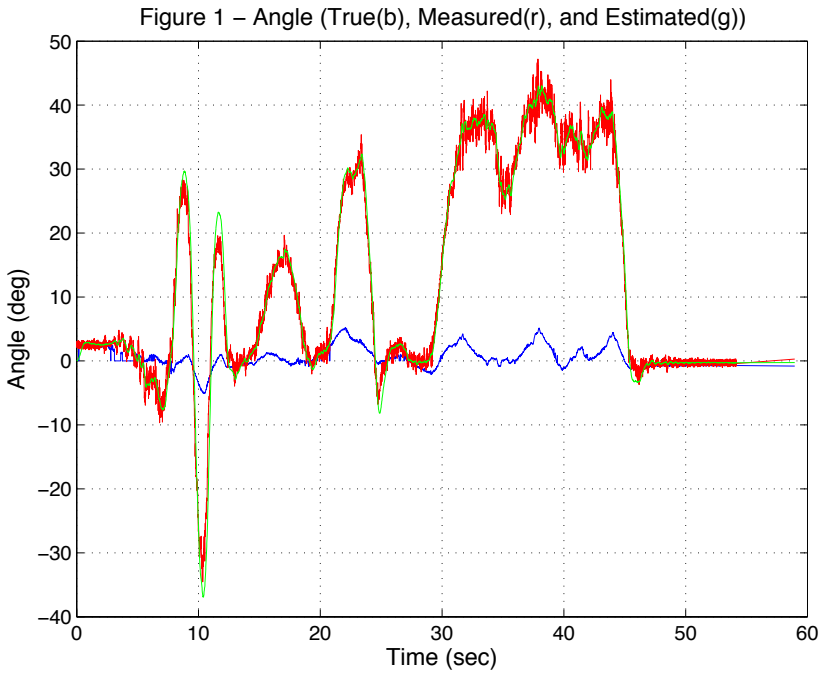

Figure 2. Results from angle estimation using Kalman Filter. Blue line is the Angle measured by the VBOX GPS system and can be considered true, red is the measured angle input that is based on Ax and Ay signals, and green is the estimated angle. As you can see the Kalman Filter do not take into account that the IMU is mounted in a car and will have large longitudinal and lateral accelerations. This filter would only work for a free body in space like a Quadrocopter with low dynamic motion.

[3] http://en.wikipedia.org/wiki/Kalman_filter

[4] D. Simon, Kalman filtering, *Embedded Systems Programming*, pp. 72-79, June 2001.

2.1.6.2 Introduction to Unscented Kalman Filter

The EKF is as it sounds, an extension of the KF to let it estimate nonlinear system. In order to do this the nonlinear system have to be linearized at each time step, which means that the EKF need to have the Jacobian Matrixes[5] that is the first order partial derivatives. So in the equations (26-32) the state transition and observation matrices are defined to be the following Jacobians:

State transition Jacobian

$$A_k = \frac{\partial f\left(\hat{x}_{k-1|k-1}, u_k\right)}{\partial x_{k-1|k-1}} \tag{35}$$

Observation matrix Jacobian

$$C_k = \frac{\partial h\left(\hat{x}_{k|k-1}, u_k\right)}{\partial x_{k|k-1}} \tag{36}$$

This can become computationally expensive and it can also lead to diverging states over time. To overcome the drawback of EKF other filters like the UKF was developed, there are many more filters so do not think that the UKF will solve all of your future problems. EKF is used very frequently in navigation systems and GPS.

The UKF filter use something called the Unscented Transform (UT) presented by Julier and Uhlmann[6]. In order to briefly understand what it is one could explain it in the following way. When performing the UT we will pick a number of points (sigma points) that can represent the mean and covariance of the state $\hat{x}$. Then we propagate these points through the non-linear function of our system. From this we will get "weights" that we will use in our prediction and update functions of the Unscented Kalman Filter. More information can be found in Matlab toolbox documentation[7], which describes the filter that you will be working with in this laboratory assignment. What should be noted here is that on page 23 in this documentation the parameters α, β and κ are discussed briefly and for more information on these please refer to the Uhlmann paper. These parameters could be used to fine tune the filter. There are also very good explanation of UKF on Wikipedia (English version) that will be helpful and we will not repeat the equations in this handout, so consult the references for further information.

You will have some more information about Kalman Filter, Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF)[8] in the referenced papers that you will get as handouts during the course.

2.2 Laboratory set-up

This laboratory assignment is divided into 2 parts:

1. Washout filtering approach of side-slip estimation
2. Unscented Kalman Filter

---

[5] http://en.wikipedia.org/wiki/Jacobian_matrix_and_determinant

[6] Julier S., Uhlmann J and Durrant-Whyte F. A New Method for the Nonlinear Transformation of Mean and Covariances in Filters and Estimators, *IEEE Transactions on Automatic Control*, vol. 45, no. 3, 2000.

[7] Hartikainen J and Särkkä S., Optimal filtering with Kalman filters and smoothers – a Manual for Matlab toolbox EKF/UKF, Version 1.2, 2008.

[8] Kandepu R., Foss B. and Imsland L., Applying the unscented Kalman filter for nonlinear state estimation, *Journal of process control* (2008), doi:10.1016/j.jprocont.2007.11.004.

In this assignment you will use logged data from the GPS and Inertial Measurement Unit (See Figure 3) mounted on the Volvo V40 test vehicle. This data you will use to estimate the Slip Angle of the V40 chassis during different test-drive scenarios.

All groups will work with the same data.

Figure 3. VBOX GPS and IMU for logging vehicle data.

2.2.1 The test drive scenarios

There are 5 files with logged data available for you, one file contains logged data when standing still which will come in handy when tuning the filters and finding covariance values in the optional task so this file you will not use for estimating the side slip angle of the vehicle. The rest of the files come from the tests outlined below.

2.2.1.1 Constant radius cornering

The first one is constant radius cornering where we slowly build up speed while trying to maintain a constant radius. The radius of the circle is approximately 13 m. We slowly build up speed until we have reached saturation of the tyres, e.g. when the tyres slide in the front and we are heavily understeering. Start of the recording is when we drive off and we stop the recording when we have gained control of the car and slowed down to a reasonable speed.

2.2.1.2 Sine with dwell (FMVSS126)

The second test manoeuvre is something similar to the FMVSS126 or sine with dwell test where we will steer with a sinusoidal and hold at the second peek for a period of time as depicted in Figure 4.

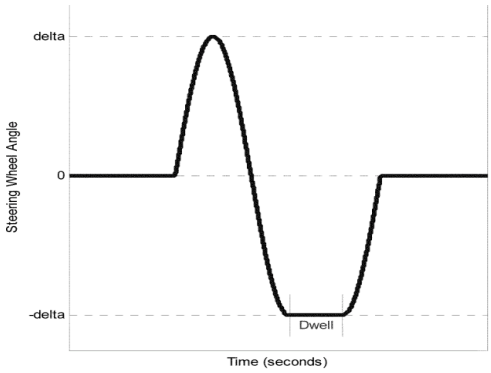

Figure 4. Sine with dwell test, FMVSS126.

The intention with this test is to get high transients and force an oversteered skid of the vehicle, meaning that the rear axle will have higher slip than the front. The speed during this test was roughly 80 km/h.

2.2.1.3 Slalom 35 km/h

The third test is slalom in relative low speed 35 km/h where we start logging when we drive off and end when we have slowed down to a low speed.

2.2.1.4 High speed step steer

The fourth test is high speed step steer where we accelerate up to 80 km/h and then steer quickly with a larger step in the steer angle and hold that angle for a while so that we will saturate the tyres ending up in an understeer situation.

2.2.2 Data to use for your filter

There are a number of signals logged from the test vehicle and the VBOX unit. In Table 1 you will see a list of the signals with description and note. Please take special care to the "Matlab syntax", the channels could change if settings in the VBOX has changes, you should always double check with the printed text in the Command Window in Matlab upon loading of the data file. The optional signals can be used to create a better process model but it is not needed to use all of them in the optional task. One can still improve the process model without additional input signals. There are of course more signals logged but these are not usually found in todays vehicles hence not available for use in you process model.

Table 1. Signals from the VBOX log system.

Variable name	Matlab syntax	Description	Note
Time	vbo.channels(1, 2).data	Time in seconds	Need to subtract first time from time vector to get a start time from zero.
rollRate_VBOX	vbo.channels(1, 40).data	deg/sec	Input variable in UKF optional, need to change sign since it is inverted
pitchRate_VBOX	vbo.channels(1, 39).data	deg/sec	Input variable in UKF optional, need to change sign since it is inverted
yawRate_VBOX	vbo.channels(1, 35).data	deg/sec	Measurement variable in UKF, need to change sign since it is inverted
vx_VBOX	vbo.channels(1, 5).data	km/h	Measurement variable in UKF
vy_VBOX	vbo.channels(1, 26).data	km/h	Only for validation and plotting
delta_VBOX	vbo.channels(1, 9).data	deg	Input variable in UKF (this is the wheel angle)
ax_VBOX	vbo.channels(1, 36).data	g	Input variable in UKF optional
ay_VBOX	vbo.channels(1, 37).data	g	Measurement variable in UKF
az_VBOX	vbo.channels(1, 41).data	g	Input variable in UKF optional
Beta_VBOX	vbo.channels(1, 30).data	deg	Only for validation and plotting, measured at CoG

2.3 Tasks to do and questions to answer

In this section you will find information about the tasks that you should do in this laboratory exercise and also the questions that you have to answer in the report. For both the washout and UKF you should show results from the Constant radius cornering, Slalom, Sine with dwell, and Step steer. In a real life implementation we need to make sure that our implementation will work in every situation.

2.3.1 Washout filtering approach of side-slip estimation

- **Task 1:** Build a Matlab/Simulink architecture that should include three body side-slip estimators; a model-based, integration of lateral acceleration based, and a washout filtered based. Illustrate your estimator architecture in the report and upload the files to bilda when handing in the report. Use the template file *Init_for_washout_filter.m* to load all the variables and vehicle parameters to the workspace.

- **Task 2:** Set and tune the vehicle parameters, see Appendix A for default values. Some vehicle parameters need tuning such as tyre parameters. The tuning is done by comparing results from your estimator design with the "truth" = body side-slip from reference measurement system (VBOX). All tasks in this section should be done by using one fix tuning (i.e. tuning should not be adapted for Tasks 3-9). Explain your tuning process.

- **Task 3:** Tune and find an appropriate value of the filter coefficient T in the wash-out filter. Motivate how you select your final tuning. You should find one value of T that is reasonably good for all tests scenarios.

- **Task 4:** Evaluate the quality of the body side-slip estimates from the three estimators and for all the test drive scenarios described in section 2.2.1 by calculating the Mean Squared Error and Max error towards the reference (which is the side-slip from VBOX), use formulation given in section 2.4.1 and that is visible in the UKF_start.m template. Make a table with the resulting MSE, and Max error for each manoeuvre.

- **Task 5:** Investigate how the three estimators tend to drift over time, how their accuracy depends on steady-state or transient manoeuvres, how accuracy is depending on vehicle speed. Explain the reasons for the estimators' tendencies.

- **Task 6:** Summaries under which circumstances it is difficult to estimate body-side slip.

- ***Extra task 1.1:*** *(for higher grade): Let the filter coefficient T be dependent on whether the vehicle is under a transient motion. The less transient, the more weight should be put on the model-based term of Equation* (13)*. Hint: Let T be a linear function of the vehicle's yaw acceleration. Show your design and the improvement in accuracy.*

- ***Extra task 1.2:*** *(for higher grade): Suggest, describe or show improvements that could make the wash-out filter even better.*

2.3.2 Unscented Kalman Filter estimation

- **Task 7:** If you study the simple Kalman Filter implementation in Section 2.1.6.1 what would you do in order to be able to use this filter in a vehicle given the input data that you have given for this exercise? You cannot change type of Kalman Filter but you can use logics and steady state vehicle kinematic models.

- We will use part of the Matlab Toolbox developed by Hartikainen and Särkkä[9], more specifically you will be using the following main functions (*ut_predict1.m* and *ut_update1.m*) and some other sub-function, which you will find in the *scripts* folder when you unzip the file.

- Start by downloading the *UKF-Template.zip* file from the Bilda webpage to your computer and then unzip it. The logged data files from VBOX is in the folder "*logged_data*". The only files that you should edit and work with is *UKF_start.m, Vehicle_state_eq.m, Vehicle_measure_eq.m.* Leave all the other files in the *scripts* folder and *logged_data* folder as they are.

- Use the equations below to design your UKF filter, not necessary to build state-space model:

$$\uparrow: m\underbrace{\left(\dot{v}_x - \dot{\psi}_z v_y\right)}_{a_x} = -F_{12}\sin(\delta) \tag{37}$$

$$\leftarrow: m\underbrace{\left(\dot{v}_y - \dot{\psi}_z v_x\right)}_{a_y} = F_{34} + F_{12}\cos(\delta) \tag{38}$$

$$I_z\ddot{\psi} = l_f F_{12}\cos(\delta) - l_r F_{34} \tag{39}$$

$$F_{12} = -C_{12}\alpha_{12} \tag{40}$$

$$F_{34} = -C_{34}\alpha_{34} \tag{41}$$

$$\alpha_{34} = \arctan\left(\frac{v_y - \dot{\psi}_z l_r}{v_x}\right) \tag{42}$$

$$\alpha_{12} = \arctan\left(\frac{v_y + \dot{\psi}_z l_f}{v_x}\right) - \delta \tag{43}$$

- The state variables should be; v_x, v_y and $\dot{\psi}_z$. The input variable should be δ. The measurement variables should be; v_x, a_y and $\dot{\psi}_z$.

- **Task 8:** Study your equations, will we have problem with observability for some values of the variables?

[9] Hartikainen J and Särkkä S., Optimal filtering with Kalman filters and smoothers – a Manual for Matlab toolbox EKF/UKF, Version 1.2, 2008.

- **Task 9:** Show for the four manoeuvres how the UKF differs from the model-based, integration of lateral acceleration based, and washout filtered based designs. Use the same tuned values of cornering stiffness as you did for the model based estimation.

- ***Extra task 2.1:*** *(for higher grade): tune your model using the alpha, beta and kappa terms of the UKF and more importantly the covariance and process noises. Show eventual improvements by comparison with original UKF design and discuss the effects in the report.*

- ***Extra task 2.2:*** *(for higher grade): extend the vehicle dynamics model for the state and measurement equations, you can choose how to extend it but you should study the parameter list given in Appendix A and the SD2225 course literature since it can give you a hint of what you can do. Show the difference in results to the base UKF and describe why the results got better or worse.*

3 Examination

For details on the grading process of this course please consult the SD2231_Grading_Criteria.pdf document on Bilda.

3.1 Report writing

A good report includes exactly the information that is needed for the reader to understand the results and nothing more. Start working with the report from the start by writing down each of the steps you have done including the approach, gathered information and the rationale of each of the steps i.e. why you decided to go in a certain direction and why that is better than another way. This will help you in the course and it will be easier to finalize the report in time.

Include the following files when submitting your report:

- Your finalized report as *pdf* or *docx*
- A zip file including all your organized Matlab and Simulink files, teachers have to be able to run your files after downloading them from Bilda.

Note: arrange your report hierarchy exactly based on the tasks arrangement in this hand-out.

Do not forget to answer the questions in this hand-out!

You should use the template on Bilda for your final report.

3.2 Evaluation code of your filter performance

In order to be able to objectively measure the performance of your filter the following process and code will be used.

We will calculate the Mean Squared Error (MSE) and the Maximum Error (MAXERR) between the true side slip value (from VBOX) and the estimated from your different estimation models. This will give you a tool to use in order to evaluate your filter designs and also to summarize and discuss the differences between the filters.

The teachers would also like to see a table of every filters MSE and MAXERR during the presentation.

The code is visible in the scripts folder and is named errorCalc.m. You can implement it in the following way:

```
%-------------------------------------------------------------
% CALCULATE THE ERROR VALES FOR THE ESTIMATE OF SLIP ANGLE
%-------------------------------------------------------------
[e_beta_mean,e_beta_max,time_at_max,error] = errorCalc(YOUR BETA, Beta_VBOX);
disp(' ');
fprintf('The MSE of Beta estimation is: %d \n',e_beta_mean);
fprintf('The Max error of Beta estimation is: %d \n',e_beta_max);
```

3.3 Teachers who will support this laboratory assignment

- Mikael Nybacka, mnybacka@kth.se, 08 - 790 76 38
- Mats Jonasson, mats.jonasson@volvocars.com

Appendix A – Vehicle parameters

Vehicle parameters:

Length from COG to front axle	l_f = 0.41*2.55 = 1.0455 m
Length from COG to front axle	l_r = 2.55 - l_f = 1.5045 m
Track width	t_w = 1.565 m
Vehicle mass	m = 1435-80 kg
Vehicle inertia	I_z = 2380 kgm^2
Gravity constant	g = 9.81 m/s^2
Tyre cornering stiffness front axle	C_{12} = 100000 N/rad
Tyre cornering stiffness rear axle	C_{34} = 100000 N/rad
Tyre radius	r_t = 0.312 m
Tyre longitudinal relaxation length	Lx_{relax} = 0.05 m
Tyre lateral relaxation length	Ly_{relax} = 0.15 m
Rolling resistance	f_r = 0.01
Steering gear ratio	K_s = 17
Vehicle roll gradient	k_{roll} = 5 deg/g
IMU (sensor) position relative COG	[r_x, r_y, r_z]=[0.4, 0, 0] m
Tyre peak value factor	D
Tyre shape factor	C
Tyre stiffness factor	B

Appendix B – Troubleshooting

Some simple tips to make your lab run smoothly.

1. If your code is slow, analyse it with the following sequence of code in the Matlab Command window:
    a. profile on
    b. profile clear
    c. "Run your code"
    d. profreport('Your code file name')

    From this you will be able to see what parts in your code that takes a lot of calculation time.
2. Use vector operator whenever possible ".* and ./ and .^ etc." instead of for loops.
3. Reduce if statements.
4. Check that your vectors are not transposed.