# CHALMERS

CHALMERS
VEHICLE
SIMULATOR

# Vehicle modelling and washout filter tuning for the Chalmers Vehicle Simulator


**Nikolce Murgovski**

Supervisor: Prof. Jonas Sjöberg
Prof. Per-Olof Gutman
Examiner: Prof. Bernhard Mehlig

**Master program:**
*Complex Adaptive Systems*
*Department of Physics*

**Master thesis developed at**:
*Control, Automation and Mechatronics Laboratory*
*Department of Signals and Systems*

CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden

# Abstract

The Chalmers Vehicle Simulator (CVS) was built by students in 1999 and is constantly being upgraded ever since. The main objective is to provide a realistic simulation environment for students to perform projects and master thesis works, investigating and testing new products and ideas relevant to the car industry. It consists of a hexapod motion platform (Stewart platform) and five computers responsible for the simulation. A quarter of a Volvo car (the part where the driver sits) is mounted on the platform and the visual cues are projected on a screen in front of the driver.

The algorithm that transforms the desired vehicle motion into realizable simulator motion commands is called a washout filter. The washout filter is responsible for keeping the motion platform within its physical boundaries and for stimulating the driver to feel that driving the simulator is close to driving a real car. The washout filter "washes out" cues below the driver's perception threshold and returns the platform state to the neutral position. It calculates the platform position and angular displacement in real time, taking the desired translational acceleration and angular displacement as an input signal. Washout filters have been widely investigated, mainly in the field of flight simulations.

In this work three washout filters ([ 1 ][ 7 ],[ 8 ], [ 9 ]) originally developed for NASA airplane simulators are considered for the CVS. The Classical and Optimal washout filters are implemented for real-time use, while the Adaptive washout filter is tested only by off-line simulation.

The quality of a washout filter depends on how realistic motion it produces. This goal is achieved by minimizing the difference between the measured sensed accelerations and rotations that a driver feels in a real car, with those the driver feels in the simulator. The sensed accelerations and rotations are determined by a mathematical model of the human vestibular system which is mainly responsible for motion sensations. An interesting feature of the vestibular system is that it does not differentiate between accelerations produced by translational movement and accelerations produced by tilting the driver's head with respect to the gravity vector. This phenomenon is known as "tilt coordination" and makes it possible to simulate low frequency translational accelerations by tilting the platform. This augments the high frequency acceleration cues produced by the washout filters.

The washout filter parameters are tuned by optimization algorithms. A Genetic Algorithm is used to find a starting point in the parameter space for the ensuing local optimization where a Riccati Algebraic Solver and the Steepest Descent Method are used. The optimization is performed on a computer simulation model of the CVS, taking standard driving manoeuvres as inputs.

The obtained Classical and Optimal washout filters were tested in real time on the CVS with several "test drivers". During all the tests, the platform never hit the physical boundaries, but moved very close to them, thus using most of the actuator's movement. According to the test drivers' suggestions some of the input signals were rescaled. After the final adjustments their impression was that the washout filters produced realistic driving experience.

# Table of Contents

**1. Introduction ... 1**
1.1. Novelty ... 2
**2. The Chalmers Vehicle Simulator (CVS) ... 3**
2.1. Components of the CVS ... 4
2.2. Stewart Platform ... 5
**3. Reference Frames and the Variables Within ... 6**
3.1. Reference Frames ... 6
*3.1.1.* Inertial Frame *I* ... 7
*3.1.2.* Simulator Frame *S* ... 7
*3.1.3.* Simulator Driver Frame *D* ... 7
*3.1.4.* Car Frame *A* ... 8
*3.1.5.* Car Driver Frame *E* ... 8
3.2. Relations between Reference Frames ... 8
3.3. Euler Angles ... 8
3.4. Translational Acceleration ... 9
3.5. Conversions of vectors between the frames ... 9
3.6. Rotation of frames ... 11
3.7. Relations between angular velocity and Euler Angles ... 11
3.8. Inertial acceleration in rotating frames ... 12
**4. Human Perception of Movement ... 13**
4.1. Sensors of movement in humans ... 13
4.2. The Vestibular System ... 14
4.2.1. Semicircular Canals ... 14
4.2.2. Otolith ... 16
4.2.3. Behaviour of the Vestibular System Model ... 17
**5. Kinematics of the Stewart Platform ... 19**
5.1. Frames of the Stewart Platform ... 19
5.2. Position and Orientation ... 19
5.3. Gimbal Positions ... 20
5.4. Actuator Lengths ... 21
**6. Motion Cuing ... 22**
6.1. Outputs of the Vehicle ... 22
6.2. The general framework of motion cuing ... 23
6.3. Specific Force and Rotation felt by the Driver ... 24
6.4. Tilt Coordination ... 26
6.5. Washout Filter Alternatives ... 28
**7. Classical Washout Filter (CWF) ... 29**
7.1. Translational Channel ... 29
7.2. Coordination Channel ... 30
7.3. Rotational Channel ... 30
7.4. Scale and Limit Block ... 31
7.5. High-pass and Low-pass Filters ... 31
7.6. CWF optimization ... 33
7.6.1. Pitch/Surge mode ... 33
7.6.2. Roll/Sway mode ... 34
7.6.3. Yaw mode ... 34
7.6.4. Heave mode ... 34
**8. Max-Tilt Washout Filter (MTWF) ... 35**
**9. Optimal Washout Filter (OWF) ... 36**
9.1. Problem structure and dynamics ... 37
9.2. Implementation of the Vestibular Model ... 40
9.3. Optimization of OWF by using EA ... 45
**10. Adaptive Washout Filter (AWF) ... 46**
10.1. Pitch/Surge Mode ... 46
10.2. Roll/Sway Mode ... 47
10.3. Yaw Mode ... 47
10.4. Heave Mode ... 48

10.5. Optimization of AWF by using EA ..... 48
**11. Nonlinear Washout Filter (NWF) ..... 49**
**12. Results ..... 50**
**13. Discussion ..... 52**
**14. Conclusion and Future Work ..... 54**
**15. References ..... 55**
**Appendix A. Vestibular System ..... 1**
**Appendix B. Stewart Platform ..... 3**
**Appendix C. Motion Cuing ..... 4**
**Appendix D. Optimization Algorithms ..... 7**
**Appendix E. Classical Washout Filter ..... 11**
**Appendix F. Optimal Washout Filter ..... 20**
**Appendix G. Adaptive Washout Filter ..... 37**

# List of Figures

*Figure 2.1: Chalmers Vehicle Simulator* ..... 3
*Figure 2.2: Block diagram of the Chalmers Vehicle Simulator* ..... 4
*Figure 2.3: Stewart Platform* ..... 5
*Figure 3.1: Reference frames in the car [ 6 ]* ..... 6
*Figure 3.2: Reference frames in the simulator [ 6 ]* ..... 6
*Figure 3.3: Overview of reference frames [ 6 ]* ..... 7
*Figure 3.4: Rotation around the x-axis* ..... 10
*Figure 3.5: Rotation around the y-axis* ..... 10
*Figure 3.6: Rotation around the z-axis* ..... 10
*Figure 3.7: Rotation around the $\omega$ axis* ..... 11
*Figure 3.8: Acceleration of a point P in the S-frame* ..... 12
*Figure 3.9: Complex rotation* ..... 12
*Figure 4.1: Interior of the otoliths and semicircular canals [ 2 ]* ..... 13
*Figure 4.2: Location of the vestibular system [ 1 ]* ..... 14
*Figure 4.3: Displacement of the cupola [ 1 ]* ..... 15
*Figure 4.4: Transfer function for the semicircular canals [ 3 ]* ..... 15
*Figure 4.5: Displacement of the Otolithic membrane due to forward acceleration [ 1 ]* ..... 16
*Figure 4.6: Transfer function for the otoliths [ 3 ]* ..... 16
*Figure 4.7: Bode plot of the semicircular canal about the x-axis* ..... 18
*Figure 4.8: Bode plot of the otolith about the x-axis* ..... 18
*Figure 5.1: Frames of the Stewart Platform [ 6 ]* ..... 19
*Figure 5.2: Calculating actuator lengths [ 6 ]* ..... 20
*Figure 6.1: General framework of motion cuing* ..... 23
*Figure 6.2: Calculation of specific ( $f_{EA}, \omega_A$ ) and sensed ( $\hat{f}_{EE}, \hat{\omega}_{EE}$ ) forces and rotation in a car* ..... 25
*Figure 6.3: Calculation of specific ( $f_{DS}, \omega_S$ ) and sensed ( $\hat{f}_{DD}, \hat{\omega}_{DD}$ ) forces and rotation in the simulator* ..... 26
*Figure 6.4: Simulated acceleration about the x-axis* ..... 27
*Figure 7.1: Classical washout filter* ..... 29
*Figure 8.1: Max-Tilt washout filter [ 6 ]* ..... 35
*Figure 9.1: Optimal washout filter* ..... 36
*Figure 9.2: Development of OWF* ..... 37
*Figure 10.1: Adaptive washout filter [ 1 ]* ..... 46
*Figure 11.1: Nonlinear Washout Filter [ 1 ]* ..... 49
*Figure 12.1: Response of the WFs on surge acceleration* ..... 51
*Figure 12.2: Response of the WFs on sway acceleration* ..... 51
*Figure 12.3: Response of the WFs on yaw displacement* ..... 51
*Figure A.1: Sensed angular velocity for the semicircular canal about the x-axis* ..... 2
*Figure A.2: Sensed force for the otolith about the x-axis* ..... 2
*Figure C.1: Sample acceleration taken from the Matlab software used to control the CVS* ..... 4
*Figure C.2: Sample angular displacement taken from the Matlab software used to control the CVS* ..... 4
*Figure C.3: Noise in longitudinal acceleration* ..... 5
*Figure C.4: Noise (zoom inside the rectangle)* ..... 6
*Figure C.5: Frequency of the noise* ..... 6
*Figure C.6: Filtered signals* ..... 6
*Figure D.1: Search space with many local optima [ 14 ]* ..... 7
*Figure D.2: Structural Neural Network for solving real time Riccati Equation [ 1 ] (page 110)* ..... 9
*Figure D.3: Convergence of the Steepest Descent method* ..... 10
*Figure D.4: Finding minimum with Steepest Descent* ..... 10
*Figure E.1: Input/output characteristics for scaling and limiting block* ..... 11
*Figure E.2: Nonlinear input scaling [ 1 ]* ..... 11
*Figure E.3: Bode plot of the translational filter W22* ..... 14
*Figure E.4: Bode plot of the coordination filter W12* ..... 14
*Figure E.5: Bode plot of the rotational filter W11* ..... 14
*Figure E.6: CWF response to surge acceleration a) ramp to step of $2m/s^2$, b) sudden acc. and brake, c) chirp, d) filtered white noise (note: s. force = specific force)* ..... 15

*Figure E.7: CWF response to sway acceleration a ) periodic lane change, b) chirp, c) filtered white noise (note: s. force = specific force)* ..... *16*
*Figure E.8: CWF response to yaw displacement* ..... *17*
*Figure E.9: Simulink model of the CWF* ..... *18*
*Figure E.10: Actual implementation of CWF in the online vehicle model* ..... *18*
*Figure E.11: Simulink model for cost estimation used by EA* ..... *19*
*Figure F.1: Bode plot of the translational filter W22* ..... *32*
*Figure F.2: Bode plot of the coordination filter W12* ..... *32*
*Figure F.3: Bode plot of the rotational filter W11* ..... *32*
*Figure F.4: OWF response to surge acceleration a) ramp to step of* $2m/s^2$*, b) sudden acc. and brake, c) chirp, d) filtered white noise (note: s. force = specific force)* ..... *33*
*Figure F.5: OWF response to sway acceleration a) periodic lane change, b) chirp, c) filtered white noise (note: s. force = specific force)* ..... *34*
*Figure F.6: OWF response to yaw displacement* ..... *35*
*Figure F.7: Simulink model of the OWF* ..... *36*
*Figure G.1: AWF response to surge acceleration a) ramp to step of* $2m/s^2$*, b) sudden acc. and brake, c) chirp, d) filtered white noise (note: s. force = specific force)* ..... *38*
*Figure G.2: OWF response to sway acceleration a) periodic lane change, b) chirp* ..... *39*
*Figure G.3: AWF response to yaw displacement* ..... *39*
*Figure G.4: Simulink model of the AWF* ..... *40*
*Figure G.5: Simulink model of the AWF block W22* ..... *41*
*Figure G.6: Simulink model of the AWF block W11* ..... *41*

# List of Tables

*Table A.1: Parameters for the semicircular canal [ 3 ]* ..... *1*
*Table A.2: Parameters for the otoliths [ 3 ]* ..... *1*
*Table B.1: Coordinates of the upper gimbals of the motion platform* ..... *3*
*Table B.2: Coordinates of the lower gimbals of the motion platform* ..... *3*
*Table B.3: Limits of the actuators of the motion platform* ..... *3*
*Table E.1: EA attributes for the CWF optimization* ..... *12*
*Table E.2: Fields in the chromosome used in EA for the CWF optimization* ..... *12*
*Table E.3: Variables used for separate modes* ..... *12*
*Table E.4: Final values for the CWF filters* ..... *13*
*Table E.5: Values for scale and limit during optimization* ..... *13*
*Table E.6: Final values for scale and limit* ..... *13*
*Table F.1: State space variables used in the development of OWF* ..... *22*
*Table F.2: EA attributes for the OWF optimization* ..... *30*
*Table F.3: Fields in the chromosome used in EA for the OWF optimization* ..... *30*
*Table F.4: Variables used for separate modes* ..... *30*
*Table F.5: Final parameters for the Riccati Solver* ..... *30*
*Table F.6: Final values for the OWF filters* ..... *31*
*Table F.7: Final values for scale and limit* ..... *31*
*Table G.1: EA attributes for the AWF optimization* ..... *37*
*Table G.2: Final values for the AWF parameters* ..... *37*
*Table G.3: Final values for scale and limit* ..... *37*

# List of Symbols

| | | |
|---|---|---|
| $a$ | Acceleration $a = \begin{bmatrix} a_x & a_y & a_z \end{bmatrix}^T$ | 9 |
| $b$ | Position of the lower actuator gimbals | 20 |
| $e$ | Sensation error for the OWF | 37 |
| $e, d, \gamma$ | Fixed gaines used in AWF | 46 |
| $f$ | Specific force | 14 |
| $\hat{f}$ | Sensed specific force | 16 |
| $g$ | Acceleration due to gravity | 26 |
| $l$ | Actuator length | 21 |
| $r_{SI}$ | Distance between the simulator and the inertial frame | 7 |
| $s$ | Laplace variable | 14 |
| $s_I$ | Displacement in the inertial frame | 46 |
| $t$ | time | 33 |
| $u$ | Position of the upper actuator gimbals | 20 |
| $u_A$ | Input signals in a car $u_A = \begin{bmatrix} a_A, \beta_A \end{bmatrix}^T$ | 23 |
| $v$ | Velocity | 22 |
| $x$ | System state vector | 36 |
| $y$ | Desired state space system output | 36 |
| $w$ | Weights used in the AWF cost function | 46 |
| $A, B, C, D, H$ | Matrices of a state space model | 14 |
| $A^{'}$ | System matrix of the standard form optimal control system | 37 |
| $A_{PS}$ | Transformation matrix for acceleration in point *P* in respect to the simulator reference frame | 12 |
| $G_\lambda, G_\delta, K_\lambda, K_\delta$ | Steepest desent constants used in AWF | 46 |
| $J$ | System cost function | 37 |
| $L_{IS}$ | Transformation matrix from the inertial frame to the simulator frame | 9 |
| $P$ | Solution of the algebraic Riccati Equation | 37 |

| | | |
|---|---|---|
| $Q, R, R_d$ | Weighting matrices in the OWF cost function | 37 |
| $R$ | radius | 22 |
| $R_1^{'}, R_2, R_{12}$ | Weighting matrices in the OWF cost function (standard form) | 37 |
| $R_S$ | Transformation matrix for angular velocity from Euler angles | 11 |
| $T_1, T_2, T_3$ | Coefficients in the semicircular canals sensation model | 40 |
| $TC$ | Tilt coordination matrix | 26 |
| $T_S$ | Transformation matrix for Euler angles from angular velocity | 11 |
| $T_L, T_a, T_S$ | Coefficients in the semicircular canals sensation model | 14 |
| $V$ | Correlation matrix | 40 |
| $W, H, G$ | Transfer function $W = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix}$ | 29 |
| $W_{11}$ | Transfer function for the translational channel | 29 |
| $W_{12}$ | Transfer function for the coordination channel | 30 |
| $W_{21}$ | Transfer function for the rotational signals coming to the translational channel | 36 |
| $W_{22}$ | Transfer function for the rotational channel | 30 |
| $\beta$ | Euler angles $\beta = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T$ | 8 |
| $\gamma$ | Filtered white noise breaking frequency | 37 |
| $\delta, \lambda$ | Time varying gains in the AWF cost function | 46 |
| $\tau_a, \tau_L, \tau_S, K$ | Coefficients in the otolith model | 16 |
| $\tau_1, \tau_2$ | Coefficients in the otolith model | 40 |
| $\rho$ | Positive scalar in the OWF cost function | 37 |
| $\omega$ | Angular velocity $\omega = \begin{bmatrix} p & q & r \end{bmatrix}^T$ | 11 |
| $\hat{\omega}$ | Sensed angular velocity | 14 |

## Subscripts

| | | |
|---|---|---|
| $(\ )_d$ | Simulator internal state space variables included in the cost function | 37 |
| $(\ )_e$ | Sensation error | 37 |
| $(\ )_n$ | White noise input states | 37 |

| | | |
|---|---|---|
| $( )_v$ | Vestibular system | 37 |
| $( )_A$ | Car reference frame | 8 |
| $( )_D$ | Simulator driver reference frame | 7 |
| $( )_E$ | Car driver reference frame | 8 |
| $( )_I$ | Inertial reference frame | 7 |
| $( )_H$ | High-pass filter | 29 |
| $( )_{OTO}$ | Otolith model | 16 |
| $( )_{PS}$ | Driver in the car | 40 |
| $( )_R$ | Rotational channel | 35 |
| $( )_{RTZ}$ | Return to zero filter | 29 |
| $( )_S$ | Simulator reference frame | 7 |
| $( )_{SC}, ( )_{SCC}$ | Semicircular canals sensation model | 14 |
| $( )_T$ | Translational channel | 29 |

## Superscripts

| | | |
|---|---|---|
| $( )^e$ | Equilibrium | 40 |
| $( )^p$ | Angular velocity in *x* direction | 40 |
| $( )^q$ | Angular velocity in *y* direction | 40 |
| $( )^r$ | Angular velocity in *z* direction | 40 |
| $( )^x$ | *x* direction | 40 |
| $( )^y$ | *y* direction | 40 |
| $( )^z$ | *z* direction | 40 |

# 1. Introduction

The algorithm that transforms the desired vehicle motion into realizable simulator motion commands is called a washout filter.The goal of the washout filter is to keep the motion platform within its boundaries and to stimulate the driver to feel that driving the simulator is close to driving a real car. The washout filter calculates the desired position and angular displacement in real time, taking the translational acceleration and angular displacement as an input signal.

Another function of the washout filter is to return the platform into its home position even when the inputs have some constant value different from zero. This is necessary so that future acceleration and rotations can be simulated successfully.

The problem rises when low frequency translational accelerations need to be simulated, because this cannot be achieved by only using the translational movement of the platform. Therefore, a tilt of the platform is used and the gravity compensates for the low frequency components of the translational acceleration. For example, when the driver accelerates in longitudinal direction the platform is slowly tilted backwards pressing the diver to the seat. The driver perceives the same effect as higher specific force acting on him when driving actual car. In this process the driver should not feel rotation and because of that the tilt rate is limited beneath the sensitivity threshold for rotation. Therefore, only sustained acceleration (low frequency) can be simulated by tilt coordination. The high frequency components are simulated by the small translational movement the platform can give (approximately 25cm). Even when tilt coordination is applied, very big accelerations cannot be obtained because of the platform limitation. Thus, the goal is to simulate realistic motion with smaller amplitude. Signals with higher amplitude will be scaled and limited to the desired interval so that the platform's actuators do not extend over their limits.

The vestibular system in humans ([ 7 ] and [ 10 ]) is mostly responsible for the feeling of motion. This system is located in the head, close to the ears and it consists of two parts, otoliths and semicircular canals. The otoliths sense acceleration (specific force) and the semicircular canals sense rotation. Otoliths are good sensors of specific force in the frequency range of [0.2 – 10] rad/s. Semicircular canals are good sensors of rotation in the frequency range of [0.2 – 2] rad/s. The vestibular system is modelled as a linear transfer function and the reference curve to which the outputs of the washout filters are optimized is generated from this model.

The washout filters are optimized by several optimization algorithms. Optimization is prepared offline, in a computer simulation, the desired parameters for the washout filters are found and then the washout filter is implemented in real use on the Chalmers Vehicle Simulator. The quality of the washout filter directly depends on the quality of the vestibular system model. There is already improved model of the vestibular system [ 1 ] than the one used in this work and one should repeat the experiments with this new model for achieving even more realistic results.

Optimization algorithms with both heuristic and functional approach are used. The Classical washout filter is optimized by Evolutionary Algorithms (EA), the Optimal washout filter is optimized by both EA and Riccati algebraic solver and the Adaptive washout filter is optimized by EA and Steepest Descent algorithm.

## *1.1. Novelty*

In this work, three washout filters, Classical Washout Filter (CWF), Optimal Washout Filter (OWF) and Adaptive Washout Filter (AWF) are considered for the Chalmers Vehicle Simulator (CVS). These washout filters were originally developed for NASA airplane simulators by Telban and Cardullo [ 1 ] and Reid and Nahon ([ 7 ], [ 8 ], [ 9 ]). Their main idea about how these WFs operate is used in this work. For example the main idea in CWF is the usage of tilt coordination for simulating sustained accelerations by using the gravity force. The main idea behind OWF is incorporating mathematical model of the human Vestibular System. The objective of AWF is to adjust the motion platform response by using feedback connections to continuously tune a set of adaptive parameters by method of steepest descent. These ideas were used as a starting point in the development of the three washout filters represented in this work. However the further growth of the washout filter development is performed independently and all three WFs are optimized to function on the Chalmers Vehicle Simulator.

Besides the functional optimization algorithms like Riccati Algebraic Solver and Steepest Descent, a heuristic optimization is introduced by the use of Evolutionary Algorithms (EA). Moreover a combination of heuristic and functional optimization is introduced for creating even more powerful optimization algorithm that will strive to the global optimum with higher speed. While Riccati Solver and Steepest Descent work very well for finding the local optima, they can easily be trapped in one of them, thus never been able to look for the global optimum. EA on the other hand gives really good results for distinguishing global optimum when the search space contains many local optima. A wise union of these algorithms can give quite optimal solution for relatively short time. Functional and heuristic approach was used for the development of OWF and AWF and its success can be inevitable seen in the results (Chapter 12) found by a computer simulation developed in Simulink, Matlab. The actual motion of the CVS was not recorded because the “real time” position of the platform could not have been obtained at the time this thesis was prepared.

The obtained Classical and Optimal washout filters were tested in real time on the CVS with several “test drivers”. During all the tests, the platform never hit the physical boundaries, but moved very close to them, thus using most of the actuator’s movement. According to the test drivers’ suggestions some of the input signals were rescaled. After the final adjustments their impression was that the washout filters produced realistic driving experience, Chapter 13.

# 2. The Chalmers Vehicle Simulator (CVS)

The Chalmers Vehicle Simulator was built by students in 1999 and is constantly being upgraded ever since. The main objective is to provide a simulation environment where new products and ideas related to the car industry can be tested before applying them to an actual car. It consists of a motion platform (Stewart platform) and five computers responsible for the simulation. A quarter of a Volvo car (the part where the driver sits) is mounted on the platform and the visual cues are projected on a screen in front of the driver.

*Figure 2.1: Chalmers Vehicle Simulator*

## 2.1. Components of the CVS

The data diagram of the CVS is given on Figure 2.2. The motion platform is connected with five computers which all have separate functionality.

- Control Computer: This is where the simulink model is developed. The model is compiled in C – code into the XPC-Target Computer.
- XPC-Target Computer: Runs the real-time system. The communication to and from the simulator goes through this computer.
- Platform Computer: Comes together with the motion platform. It shows the status of the platform in real time and from this computer the platform can be run manually.
- Graphics Computer: The virtual environment is run on this computer.
- Sound Computer: Handles the sound of the engine.

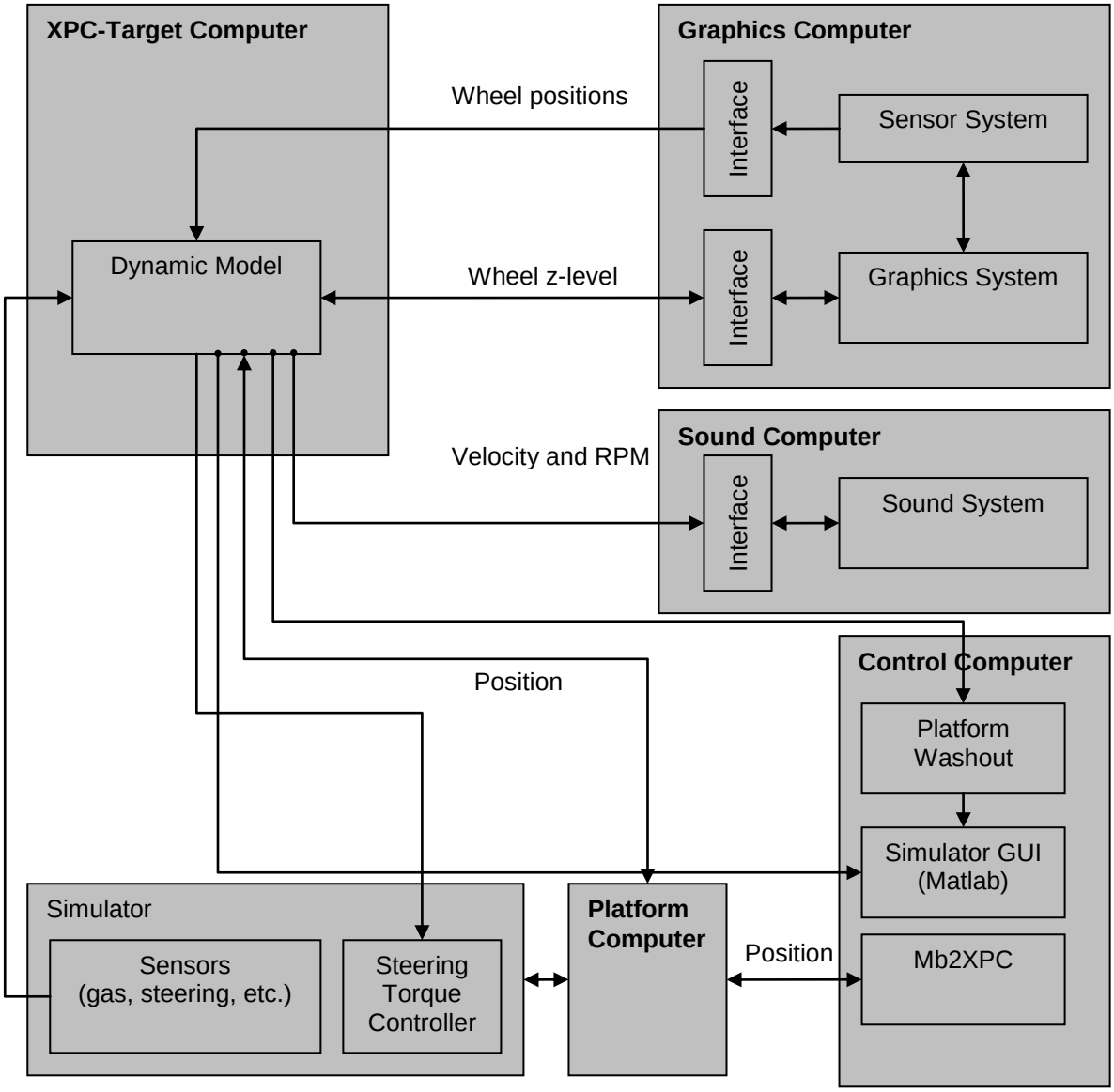


*Figure 2.2: Block diagram of the Chalmers Vehicle Simulator*

## 2.2. Stewart Platform

The Stewart Platform consists of 6 actuators which can extend or contract and 6 motors attached to them which change their angle. The platform provides motion in 6 degrees of freedom, 3 translational and 3 rotational. The motion of the platform is limited and it can move approximately 25cm in translational direction and even less when it is tilted. The maximum tilting angles are limited to about $20^{\circ}$. The actual actuators limits are given in Appendix B. The kinematics of the platform is described in Chapter 5.

*Figure 2.3: Stewart Platform*

# 3. Reference Frames and the Variables Within

This chapter introduces the reference frames used in CVS, the conversion of variables between the frames, rotation of frames, relations between angular velocity and Euler angles and inertial acceleration in rotating frames. This chapter is referenced to the work of Etkin [ 12 ], but the notation which will be used throughout the development of the washout filters is preserved as same as the notation of the previously developed washout filters [ 6 ].

## *3.1. Reference Frames*

In order to describe the development of motion drive algorithms easily, six reference frames are introduced in this research [ 6 ]. Three of these frames are connected with the real car, and the other three with the simulator (Figure 3.1- Figure 3.3). All of the frames are right handed *X x Y = Z*.

The variables are introduced by an index indicating the frame. For example, the acceleration in the inertial frame is introduced as $a_I$ .

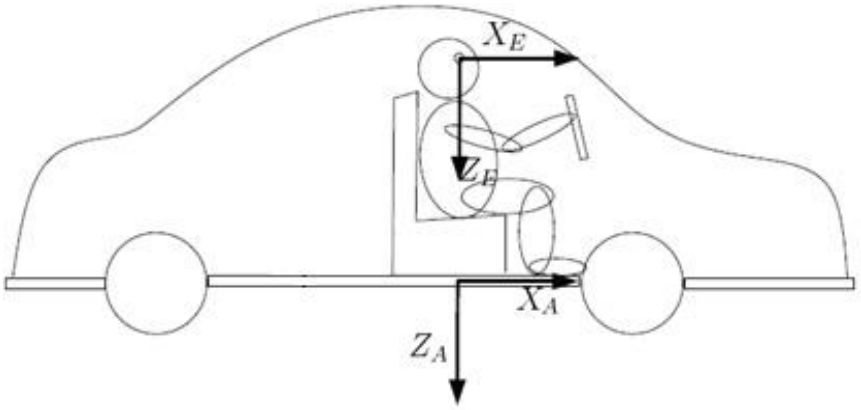


*Figure 3.1: Reference frames in the car [ 6 ]*

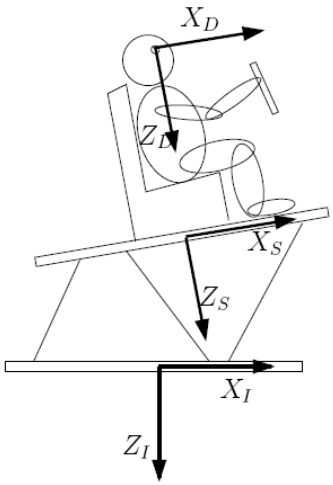


*Figure 3.2: Reference frames in the simulator [ 6 ]*

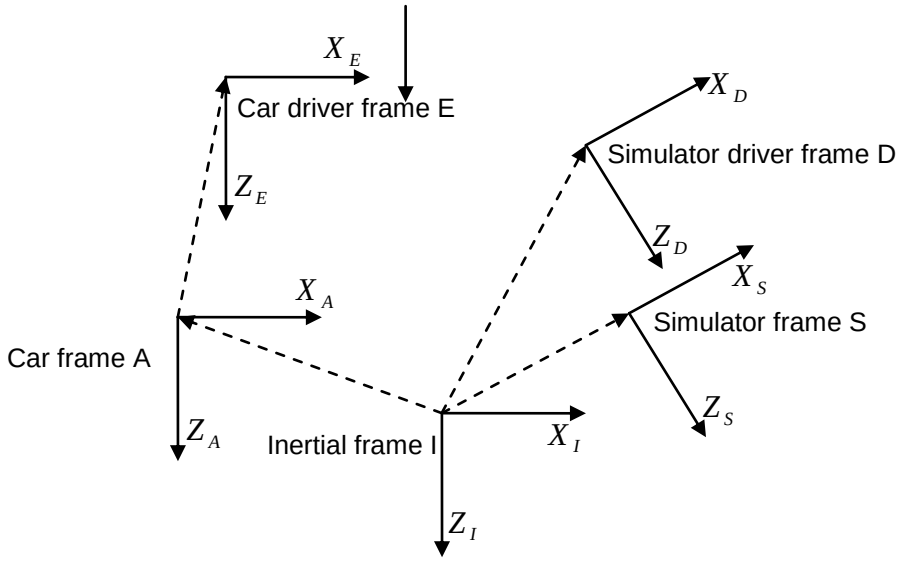


*Figure 3.3: Overview of reference frames [ 6 ]*

### *3.1.1.* Inertial Frame *I*

This frame is attached to the Earth where it stays fixed. Its position and orientation does not change over time. The *X* axis of this frame points forward, the *Y* axis points to the reader and the *Z* axis points downward. All other axes are always expressed in respect to the inertial frame, thus $X_{II}$ can be written as $X_I$ by omitting the second index. The origin of this frame is located beneath the simulator, right in the middle between the lower gimbals positions, thus we can say it belongs to the simulator. Yet, this frame is used from the real car too, because the car motion is observed from the origin of the inertial frame.

### *3.1.2.* Simulator Frame *S*

The simulator frame *S* is attached to the Stewart Platform. Its origin is placed in the middle between the upper gimbals of the platform. The position of this frame is changing over time and its origin is changing too in respect to the inertial frame. The position of its origin can be expressed at any moment by knowing the vector $r_{SI}$, which means position of the origin of frame *S* in respect to the origin of the inertial frame. The orientation of the axes is the same as the orientation of the axes in the inertial frame.

### *3.1.3.* Simulator Driver Frame *D*

The simulator driver frame *D* is parallel to the frame *S* and its origin is at the centre of the driver's head. The position of this frame is fixed in respect to the frame *S*, but it is changing over time in respect to the inertial frame. The main reason for the existence of this frame is to get the specific force and rotation at the driver's head, which is one of the main issues for building successful washout filter. This is described in details in Chapter 4.
The position of this frame's origin can be retrieved by knowing the position of the origin of the frame *S* and the vector $r_{DS}$.

### *3.1.4.* Car Frame *A*

The car frame is attached to the car. It is parallel to the base frame and its origin can be found by $r_{AI}$ in respect to the inertial frame. This frame has the same orientation with respect to the inertial frame like the simulator frame *S.*

### *3.1.5.* Car Driver Frame *E*

The origin of this frame is at the centre of the driver's head in the car. Its orientation with respect to the frame *A* is the same as the orientation of the simulator driver frame *D* in respect to the simulator frame *S*. The origin of this frame over time can be found by knowing the origin of the frame *A* and the vector $r_{EA}$ .

## *3.2. Relations between Reference Frames*

Several relations exist between the reference frames:

- Frames *A* and *E* are always parallel to each other and their relative location $r_{EA}$ is fixed all the time.
- Frames *S* and *D* are always parallel to each other and the relative location of frame *D* in respect to frame *S,* $r_{DS}$ , is fixed all the time.
- The inertial frame is fixed all the time, and its positions and origin are not changing.
- The position of frame *S* and its orientation vary over time in respect to the inertial frame. The origin of frame *S* can be found by $r_{SI}$ and the orientation is determined by the Euler angles in respect to the inertial frame.
- The position of frame *A* and its orientation vary over time in respect to the inertial frame. The origin of this frame can by found by knowing the vector $r_{AI}$ , and the position of the axes is determined by Euler angles.

## *3.3. Euler Angles*

Orientation of one frame is given with its Euler angles, relatively to another frame. In the simulator the orientation of any frame is always given in respect to the inertial frame. For example, the orientation of the platform *S*, is given by:

$$\beta_S = \begin{bmatrix} \phi_S \\ \theta_S \\ \psi_S \end{bmatrix} \qquad (3.1)$$

To obtain the final position of frame *S* relatively to frame *I*, one needs to know its orientation represented by $\beta_S$ and the coordinates represented by the vector $r_{SI}$ . The position is obtained in five steps:

- Set the initial position of frame *S* equal to frame I
- Rotate the frame *S* with angle $\phi_{SI}$ (roll) around the $X_S$ -axis
- Rotate the frame *S* with angle $\theta_{SI}$ (pitch) around the $Y_S$ -axis
- Rotate the frame *S* with angle $\psi_{SI}$ (yaw) around the $Z_S$ -axis
- Translate the frame *S* by $r_{SI}$ .

Since the Euler angles are always defined with respect to the inertial frame, the index *I* is usually dropped.

## 3.4. Translational Acceleration

The translational acceleration of the platform is given by:

$$a = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \qquad (3.2)$$

These acceleration components can be found in practise with different names:

- $a_x$ - surge (longitudinal acceleration)
- $a_y$ - sway (lateral acceleration)
- $a_z$ - heave (vertical acceleration)

In this work the notation surge, sway and heave acceleration is rather used.

## 3.5. Conversions of vectors between the frames

When the acceleration in frame *I* is known, usually there is a need to express that acceleration in the simulator frame *S* too and vice versa. If the acceleration in frame *I* is given as a vector:

$$a_I = \begin{bmatrix} a_{x_I} \\ a_{y_I} \\ a_{z_I} \end{bmatrix} \qquad (3.3)$$

then the acceleration in the *S* frame will be found as:

$$a_{SI} = L_{IS} a_I \qquad (3.4)$$

where $L_{IS}$ is three by three matrix and it is called a rotation matrix. In order to find out the rotation matrix, the frames will be rotated step by step, first around the x-axis, then around the y-axis and at last around the z-axis.

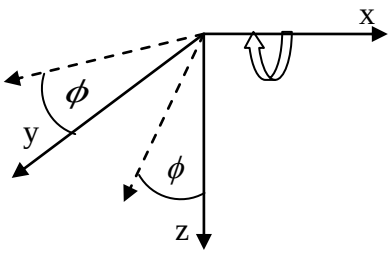


Figure 3.4: Rotation around the x-axis

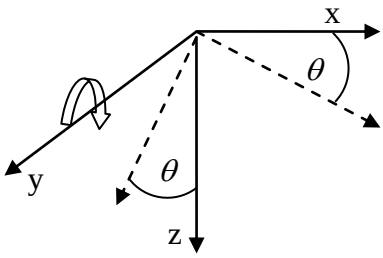


Figure 3.5: Rotation around the y-axis

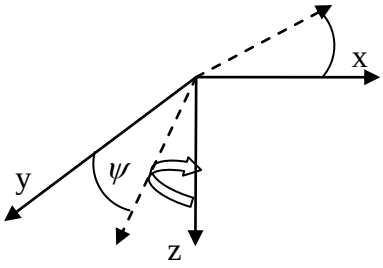


Figure 3.6: Rotation around the z-axis

These three rotations separately can be represented with the rotation matrices $R_x$, $R_y$ and $R_z$:

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix} R_y = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} R_z = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.5)$$

The rotation matrix consists of all these three rotations executed one by one and it can be represented by their product:

$$L_{IS} = R_x R_y R_z \quad (3.6)$$

$$L_{IS} = \begin{bmatrix} \cos(\theta)\cos(\psi) & \cos(\theta)\sin(\psi) & -\sin(\theta) \\ \sin(\phi)\sin(\theta)\cos(\psi) - \cos(\phi)\sin(\psi) & \sin(\phi)\sin(\theta)\sin(\psi) + \cos(\phi)\cos(\psi) & \sin(\phi)\cos(\theta) \\ \cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi) & \cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi) & \cos(\phi)\cos(\theta) \end{bmatrix} \quad (3.7)$$

The inverse operation is:

$$a_I = L_{SI} a_{SI} \quad (3.8)$$

where:

$$L_{SI} = L_{IS}^{-1} = L_{IS}^{T} \quad (3.9)$$

When converting positions from frame *I* to frame *S*, the distance between the frames ($r_{SI}$) should be added:

$$r_{SS} = r_S = L_{IS} r_I + r_{SI} \quad (3.10)$$

## 3.6. Rotation of frames

The rotation in one frame can be expressed into other by its axis of rotation. The rotation in frame *S* (Figure 3.7) is defined by rotation around the axis $\omega$, which can be divided into three components on the x, y and z axes.

$$\omega = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.11}$$

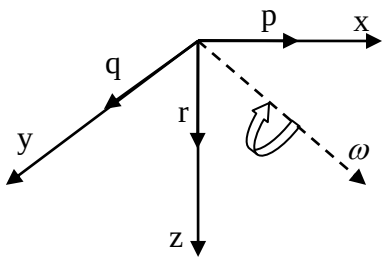


*Figure 3.7: Rotation around the $\omega$ axis*

## 3.7. Relations between angular velocity and Euler Angles

The angular velocity $\omega$ can be expressed by the Euler angles $\beta$. If we need to find the angular velocity $\omega_S$ in the *S* frame, when the Euler angles in the inertial frame are known, we can use the equation:

$$\omega_S = R_S \dot{\beta}_S \tag{3.12}$$

The transformation matrix $R_S$ is calculated step by step in the following equations:

$$\begin{bmatrix} p_S \\ q_S \\ r_S \end{bmatrix} = \begin{bmatrix} \dot{\phi}_S \\ 0 \\ 0 \end{bmatrix} + R_x \begin{bmatrix} 0 \\ \dot{\theta}_S \\ 0 \end{bmatrix} + R_x R_y \begin{bmatrix} 0 \\ 0 \\ \dot{\psi}_S \end{bmatrix} = \begin{bmatrix} \dot{\phi}_S - \dot{\psi}_S \sin(\theta_S) \\ \dot{\theta}_S \cos(\phi_S) + \dot{\psi}_S \sin(\phi_S)\cos(\theta_S) \\ \dot{\psi}_S \cos(\phi_S)\cos(\theta_S) - \dot{\theta}_S \sin(\phi_S) \end{bmatrix} = R_S \begin{bmatrix} \dot{\phi}_S \\ \dot{\theta}_S \\ \dot{\psi}_S \end{bmatrix} \tag{3.13}$$

$$R_S = \begin{bmatrix} 1 & 0 & -\sin(\theta_S) \\ 0 & \cos(\phi_S) & \sin(\phi_S)\cos(\theta_S) \\ 0 & -\sin(\phi_S) & \cos(\phi_S)\cos(\theta_S) \end{bmatrix} \tag{3.14}$$

where $R_x$, $R_y$ and $R_z$ are the rotation matrices found above, in Equation ( 3.5 ).

In the same way $\dot{\beta}_S$ can be calculated if $\omega_S$ is known by finding the inverse matrix $T_S = R_S^{-1}$:

$$\dot{\beta}_S = T_S \omega_S \tag{3.15}$$

$$T_S = \begin{bmatrix} 1 & \sin(\phi_S)\tan(\theta_S) & \cos(\phi_S)\tan(\theta_S) \\ 0 & \cos(\phi_S) & -\sin(\phi_S) \\ 0 & \sin(\phi_S)\sec(\theta_S) & \cos(\phi_S)\sec(\theta_S) \end{bmatrix} \tag{3.16}$$

## 3.8. Inertial acceleration in rotating frames

On Figure 3.8, a frame is shown which is moving with respect to the reference frame. In our case it can be considered that the reference frame is the inertial frame *I* and the moving frame in Figure 3.8 is the platform frame *S*. If a point *P* is moving around the $\omega_{SS}$-axis, then the acceleration $a_{PS}$ in respect to the reference frame can be expressed by:

$$a_{PS} = a_{SS} + \dot{\omega}_{SS} \times R_S + \omega_{SS} \times (\omega_{SS} \times R_S) \qquad (3.17)$$

where $a_S$ is the acceleration defined in *S* frame coordinates and can be calculated as:

$$a_{SS} = L_{IS} a_{SI} \qquad (3.18)$$

The angular velocity $\omega_{SS}$ can be represented by its three coordinates $\omega_{SS} = [p_{SS} \quad q_{SS} \quad r_{SS}]^T$ (Figure 3.9), but the vector representation is insufficient to hold the necessary information about the angular velocity, therefore a matrix representation is used:

$$\omega_{SS} = \begin{bmatrix} 0 & -r_{SS} & q_{SS} \\ r_{SS} & 0 & -p_{SS} \\ -q_{SS} & p_{SS} & 0 \end{bmatrix} \qquad (3.19)$$

The general equation for the inertial acceleration becomes:

$$a_{PS} = a_{SS} + A_{PS} R_S \qquad (3.20)$$

where :

$$A_{PS} = \begin{bmatrix} -q_{SS}^2 - r_{SS}^2 & p_{SS}q_{SS} - \dot{r}_{SS} & p_{SS}r_{SS} + \dot{q}_{SS} \\ p_{SS}q_{SS} + \dot{r}_{SS} & -p_{SS}^2 - r_{SS}^2 & q_{SS}r_{SS} - \dot{p}_{SS} \\ p_{SS}r_{SS} - \dot{q}_{SS} & q_{SS}r_{SS} + \dot{p}_{SS} & -p_{SS}^2 - q_{SS}^2 \end{bmatrix} \qquad (3.21)$$

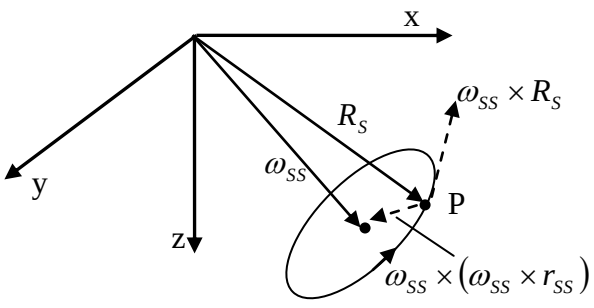


*Figure 3.8: Acceleration of a point P in the S-frame*

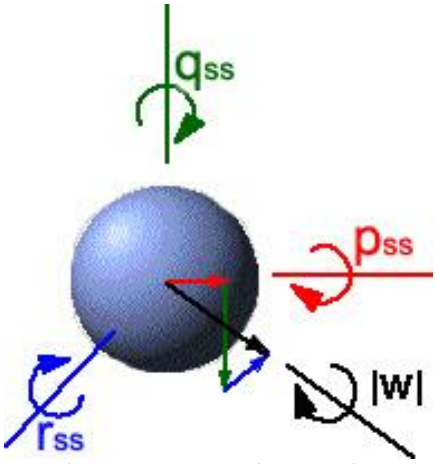


*Figure 3.9: Complex rotation*
http://www.euclideanspace.com/physics/kinematics

# 4. Human Perception of Movement

This chapter goes through the human biology and represents mathematical model of the vestibular system which is one of the sensory inputs for movement. The problem of creating good washout filter collides with the references to which the outputs are compared. One important rule that should not be broken is keeping the actuators in their limits. This will only prevent physical damage and does not help us create realistic movement. The only way of making realistic movement is making the output similar to the reference curve created by the human real perception of movement.

This chapter describes the human perception of movement, the vestibular system and its mathematical model used in this research.

## *4.1. Sensors of movement in humans*

The basic mechanisms with which human beings sense movement are:

- perception through the vestibular system
- visual perception and
- perception through distribution of proprioceptors throughout the body

The proprioceptors mainly detect joint positions and muscle effort. The joint positions could be influenced by motion when certain parts of the body are driven because of the vehicle acceleration (for example the part of the body which is in contact with the seat) and the other parts stay behind due to the inertia. The muscle effort then tends to minimize the undesired relative movement in the body. The motion can be further sensed by the touch sensors in the skin, because of the changed pattern in the contact between the body and its immediate environment [ 5 ].

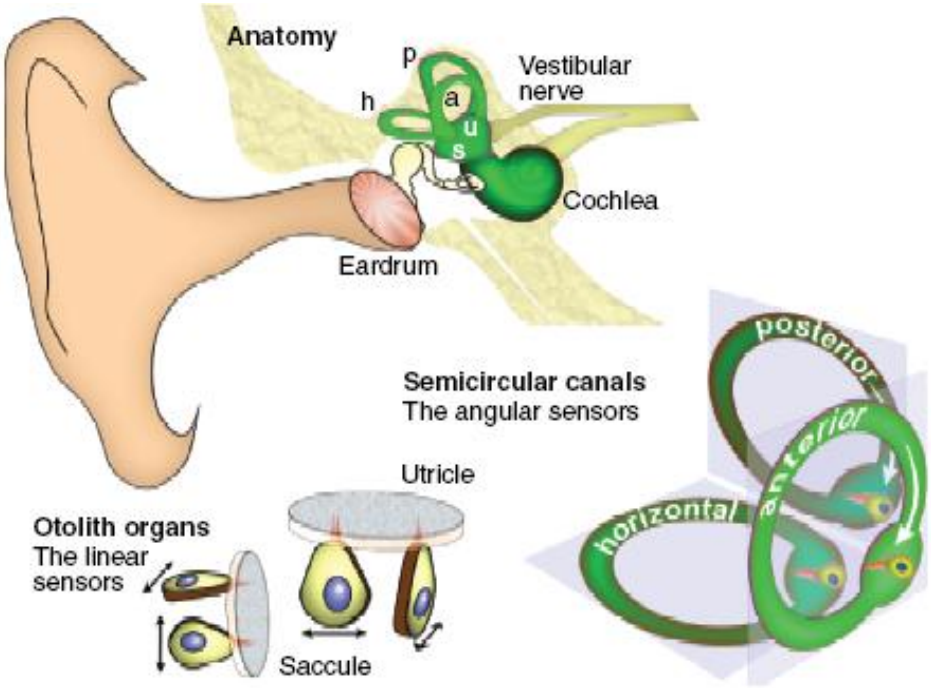


*Figure 4.1: Interior of the otoliths and semicircular canals [ 2 ]*

The visual perception can be understood when a movement is observed relatively to other object. For example, a person looking throughout the window of a noiseless train which moves with constant speed is aware of his movement because of the environment. But this system could be fooled too. For example, when two trains stand one by each other and a person from the first train looks through the window to the second train which is starting to move, he could easily get impression that the other train is staying and his train is moving in opposite direction. This trick could be also used in vehicle simulation.

In the vestibular system, two sensory parts, semicircular canals and otoliths (Figure 4.1), detect angular and linear acceleration. This system is of our main interest because it senses specific forces which can be easily produced in simulated environment.

## *4.2. The Vestibular System*

The vestibular system is located in the inner ear and consists of semicircular canals and otoliths organs that sense angular and linear motion respectively. The location and orientation of the vestibular system is shown on Figure 4.2. This sensory system is sensitive to specific forces rather than true acceleration. The specific force has the same units as acceleration, because it is the force acting on the body per unit mass of the body and it can be calculated from the acceleration when excluding the gravity:

$$f = a - g \qquad (4.1)$$

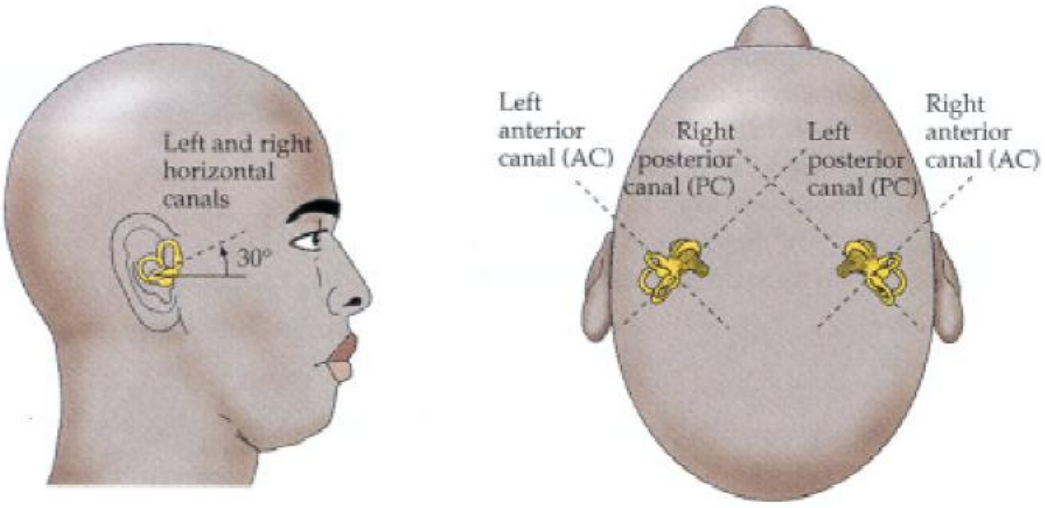


*Figure 4.2: Location of the vestibular system [ 1 ]*

### 4.2.1. Semicircular Canals

The semicircular canals sense rotational movement in space. They consist of two sets of three elliptical cavities or canals (horizontal, posterior and anterior, Figure 4.2), which are filled with liquid called endolymph. In each canal a seal-form bulb exists, called cupola, through which the endolymph cannot circulate. When the head turns in the plane of one of the canals, a force is produced due to the inertia of the endolymph and displaces the cupola in opposite direction of the head movement (Figure 4.3).

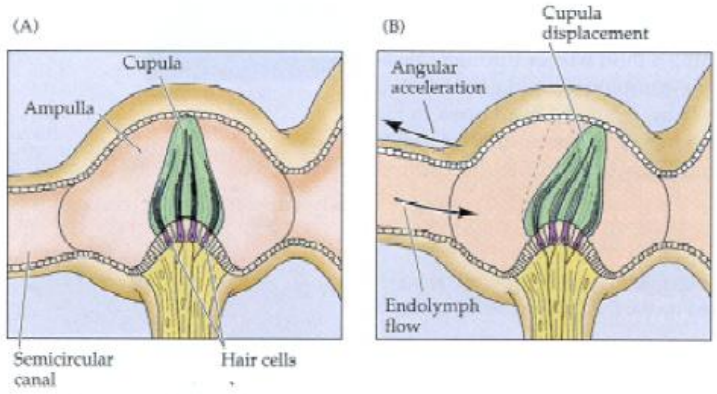


*Figure 4.3: Displacement of the cupola [ 1 ]*

This system can be represented as over-damped torsion-pendulum model with the transfer function given in Figure 4.4.

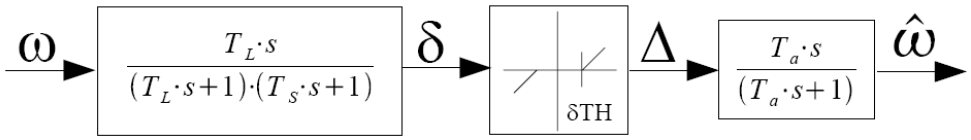


*Figure 4.4: Transfer function for the semicircular canals [ 3 ]*

The threshold $\delta TH$ comes from the fact that the humans are not sensitive to small angular velocities. However, this block is not essential in the model used in this research. Therefore, it is omitted and our assumed driver has the sixth sense to feel these small angular velocities, too. The transfer function for the system now becomes:

$$H_{sc} = \frac{T_L T_a s^2}{(T_a s+1)(T_L s+1)(T_S s+1)} = \frac{\hat{\omega}}{\omega} \qquad (4.2)$$

$$H_{sc} = \frac{\frac{1}{T_S} s^2}{s^3 + \left(\frac{1}{T_a}+\frac{1}{T_L}+\frac{1}{T_S}\right)s^2 + \left(\frac{1}{T_L T_S}+\frac{1}{T_L T_a}+\frac{1}{T_a T_S}\right)s + \frac{1}{T_a T_L T_S}} \qquad (4.3)$$

where $\hat{\omega}$ is the sensed angular velocity.

The transfer function has different coefficients for the three linear accelerations (surge, sway, and heave) and three angular accelerations (roll, pitch and yaw). The coefficients are shown in Table A.1.

The state-space differential equation corresponding to Equation ( 4.2 ) can be written as [ 10 ]:

$$\dot{\underline{x}} = \underline{A}\underline{x} + \underline{B}\omega \qquad (4.4)$$

$$\hat{\omega} = \underline{C}\underline{x}\ ,\ \underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad (4.5)$$

where

$$\underline{A} = \begin{bmatrix} -\frac{1}{T_L}-\frac{1}{T_S}-\frac{1}{T_a} & 1 & 0 \\ -\frac{1}{T_L T_S}-\frac{1}{T_L T_a}-\frac{1}{T_a T_S} & 0 & 1 \\ -\frac{1}{T_a T_L T_S} & 0 & 0 \end{bmatrix}, \underline{B} = \begin{bmatrix} \frac{1}{T_S} \\ 0 \\ 0 \end{bmatrix}, \underline{C} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \qquad (4.6)$$

In time domain the system can be represented as:

$$\dddot{\hat{\omega}} + \left(\frac{1}{T_a}+\frac{1}{T_L}+\frac{1}{T_S}\right)\ddot{\hat{\omega}} + \left(\frac{1}{T_L T_S}+\frac{1}{T_L T_a}+\frac{1}{T_a T_S}\right)\dot{\hat{\omega}} + \frac{1}{T_a T_L T_S}\hat{\omega} = \frac{1}{T_S}\ddot{\omega} \qquad (4.7)$$

## 4.2.2. Otolith

These organs are responsible for sensation of linear motion. They can sense both linear acceleration and tilting of the head in respect to the gravity vector. There are two on each side of the head, Utriccle and Saccule (Figure 4.1). The Utricle senses motion in horizontal plane and the Saccule senses motion in vertical plane. The measuring system is based on the displacement of the otolithic membrane due to linear acceleration (Figure 4.5).

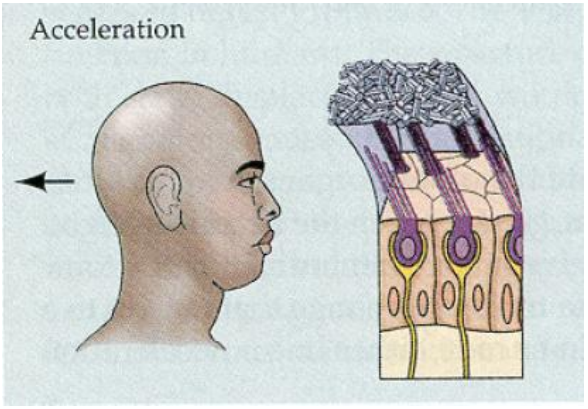


*Figure 4.5: Displacement of the Otolithic membrane due to forward acceleration [ 1 ]*

The interesting thing about the otoliths is sensation of gravity in the same way as linear acceleration. This feature opens the opportunity to use gravitational force, by tilting the platform, for simulating linear acceleration. The specific force it senses can be introduced as:

$$f = a - g \qquad (4.8)$$

The mathematical model of the otoliths is presented with the linear transfer function on Figure 4.6. Typical values for the parameters are shown on Table A.2.

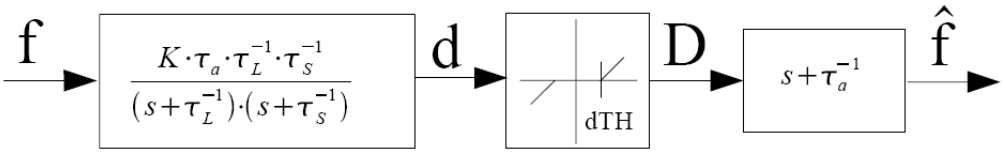


*Figure 4.6: Transfer function for the otoliths [ 3 ]*

The transfer function without the dead zone threshold can be written as:

$$H_{oto} = \frac{K(\tau_a s + 1)}{(\tau_L s + 1)(\tau_S s + 1)} = \frac{\hat{f}}{f} \qquad (4.9)$$

$$H_{oto} = \frac{\frac{K\tau_a}{\tau_L \tau_S} s + \frac{K\tau_a}{\tau_L \tau_S}}{s^2 + \left(\frac{1}{\tau_L} + \frac{1}{\tau_S}\right) s + \frac{1}{\tau_L \tau_S}} \qquad (4.10)$$

where $\hat{f}$ is the sensed force.

The state-space differential equation corresponding to Equation ( 4.9 ) can be written as: [ 10 ]

$$\dot{\underline{x}} = \underline{A}\underline{x} + \underline{B}f \qquad (4.11)$$

$$\hat{f} = \underline{C}\underline{x}, \ \underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad (4.12)$$

where

$$\underline{A} = \begin{bmatrix} -\frac{1}{\tau_L} - \frac{1}{\tau_S} & 1 \\ -\frac{1}{\tau_L \tau_S} & 0 \end{bmatrix}, \ \underline{B} = \begin{bmatrix} \frac{K\tau_a}{\tau_L \tau_S} \\ \frac{K}{\tau_L \tau_S} \end{bmatrix}, \ \underline{C} = \begin{bmatrix} 1 & 0 \end{bmatrix} \qquad (4.13)$$

In time domain the system can be represented as:

$$\dddot{\hat{f}} + \left(\frac{1}{\tau_L} + \frac{1}{\tau_S}\right) \ddot{\hat{f}} + \frac{1}{\tau_L \tau_S} \dot{\hat{f}} = \frac{K\tau_a}{\tau_L \tau_S} \ddot{f} + \frac{K\tau_a}{\tau_L \tau_S} \dot{f} \qquad (4.14)$$

### 4.2.3. Behaviour of the Vestibular System Model

The behaviour of the vestibular system model can be understood by the diagrams shown on the next two figures. On Figure 4.7 and Figure 4.8, a bode plot is shown for the semicircular canal and the otoliths. The parameters are chosen about the x-axis. It can be seen that the system is good sensor only in defined interval [ 10 ] of frequency values. According to Figure 4.7 the system is good sensor of angular velocity in the frequency band:

$$0.2 \le \omega \le 10 [\frac{rad}{s}] \qquad (4.15)$$

and looking on Figure 4.8 the system is good sensor of specific force in the frequency band

$$0.2 \le \omega \le 2 [\frac{rad}{s}] \qquad (4.16)$$

(Note 1: The low frequency sensitivity that can be seen on Figure 4.8 is irrelevant because we can not simulate low frequency acceleration by moving the platform translationally. This is achieved by using Tilt Coordination (Section 6.4). Each WF uses high pass filter in the translational channel to take away the low frequency components.
Note 2: The dead zone threshold was omitted during the bode plots.)

On Figure A.1 and Figure A.2 the sensed signals are shown for the input presented in the same figures. The simulation is made in Simulink, Matlab, for 10 time units with step size of 0.01.

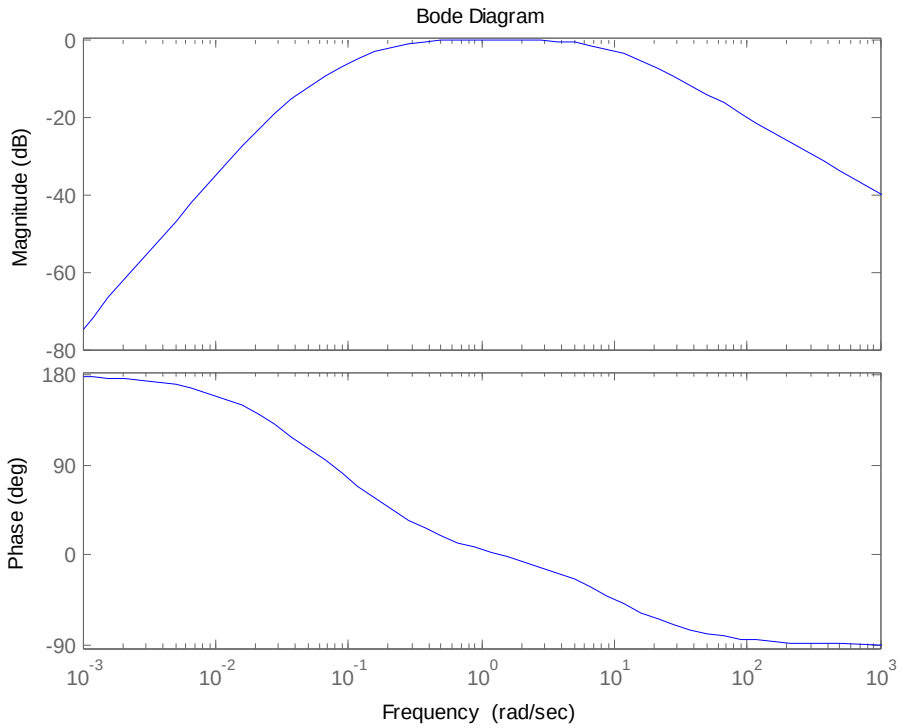


*Figure 4.7: Bode plot of the semicircular canal about the x-axis*

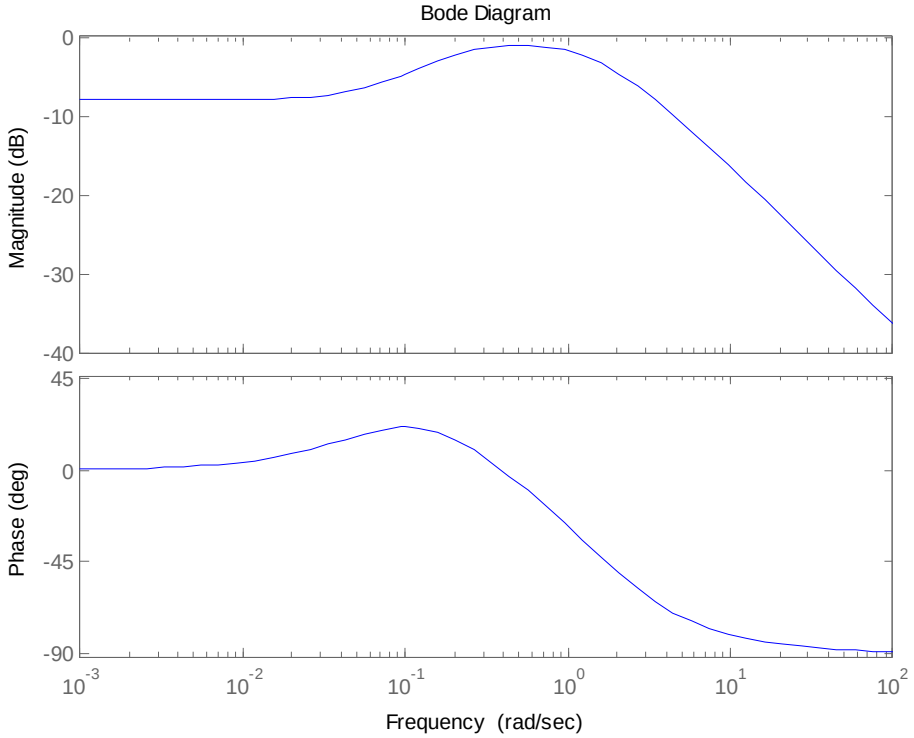


*Figure 4.8: Bode plot of the otolith about the x-axis*

# 5. Kinematics of the Stewart Platform

The base rule that needs to be preserved when moving the platform is keeping the actuators within their limits. This constraint implies calculation of the actuators lengths at each time step.

This chapter deals with the parameters of the Stewart Platform used in the CVS and the calculation of the actuator lengths. The content of the chapter is based on the work of Pieter van Balen [ 6 ].

## 5.1. Frames of the Stewart Platform

The simulator frames are shown on Figure 5.1. The inertial frame *I* is attached to the earth and this frame is not moving during the simulation. The *S* frame is a moving frame, it is attached to the payload platform of the simulator and its rest position is exactly above the inertial frame. The origin of this frame, as same as the origin of the inertial frame, is in the middle of the upper gimbal positions.

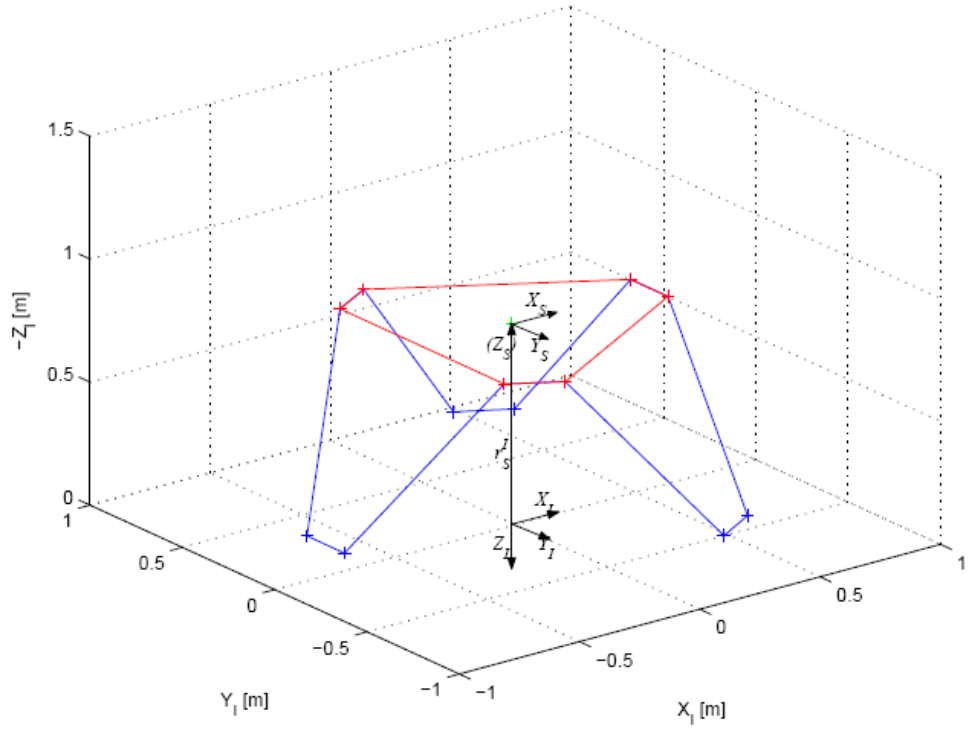


*Figure 5.1: Frames of the Stewart Platform [ 6 ]*

## 5.2. Position and Orientation

The position of the platform is represented by the vector $r_{SI}$, indicated on Figure 5.1. Its value at time *t* can be calculated as:

$$r_{SI} = r_{SI}(0) + \int_0^t \int_0^t a_{SI}(t)dt^2 \qquad (5.1)$$

where $r_{SI}(0)$ is the starting position of the platform with values:

$$r_{SI}(0) = \begin{bmatrix} 0 \\ 0 \\ -0.2 \end{bmatrix} \qquad (5.2)$$

## 5.3. Gimbal Positions

The platform has six actuators each with upper and lower gimbal. The positions of the lower gimbals are represented by $b_{1I}, b_{2I}, ..., b_{6I}$ and they are constants in frame *I*. The positions of the upper gimbals are $u_{1S}, u_{2S}, ..., u_{6S}$ and these values are constants in the frame *S*. The upper positions can be transferred in the *I* frame by:

$$u_I = L_{SI} u_S + r_{SI} \qquad (5.3)$$

where the transformation matrix $L_{SI}$ depends of the Euler angles of the platform $\beta_S$.
The coordinates of the lower and the upper gimbal are listed in Table B.1 and Table B.2 respectively.

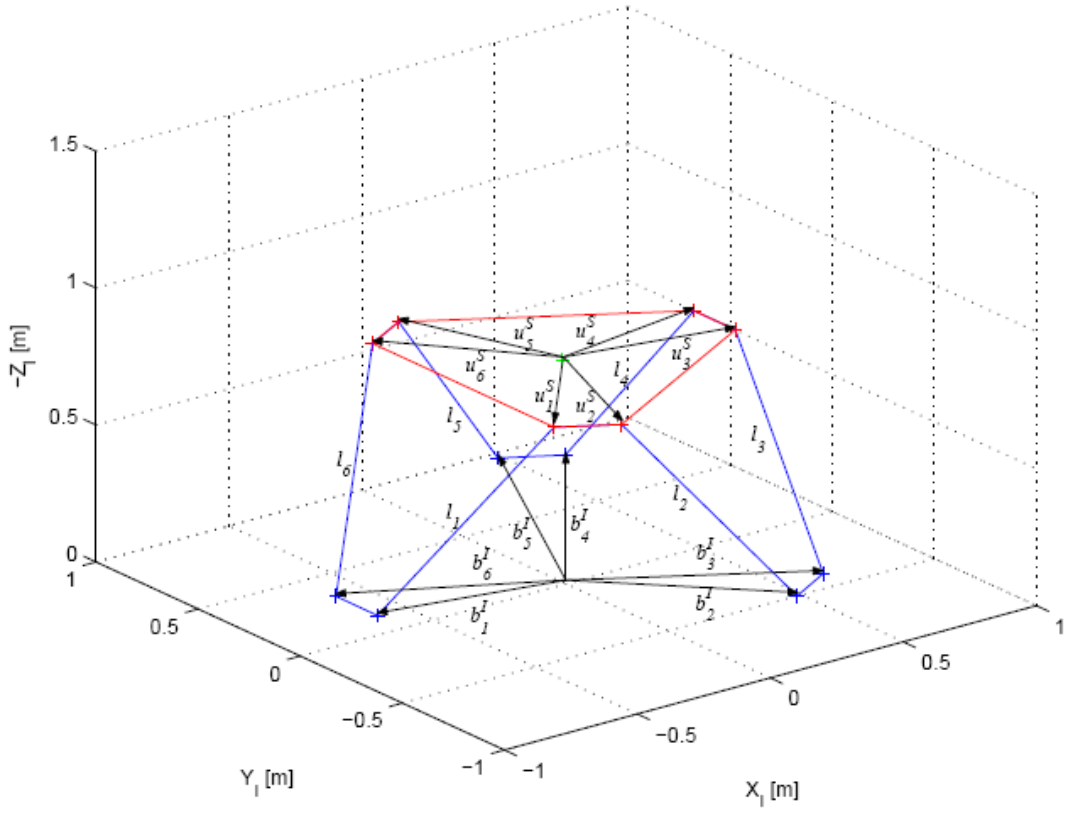


*Figure 5.2: Calculating actuator lengths [ 6 ]*

## 5.4. Actuator Lengths

The lengths of the actuators can be calculated as:

$$l = \left| u_I - b_I \right| \qquad (5.4)$$

These lengths are limited by several maximum and minimum values (Table B.1, Table B.2 and Table B.3). Firstly they are limited by software. If the actuators exceed the software limits they are mechanically cushioned and the physical limits are reached.

# 6. Motion Cuing

The motion cuing chapter deals with the basic problem in this research: How to create a simulated movement, as realistic as possible. First we need to approximate an interval in which the input values vary and this can be done without difficulty by examining a real vehicle model. The second part is creating desired output, so that the driver experiences the motion realistically, by keeping in mind not to exceed the actuator's lengths. The idea is to compare the output of the vestibular system for different input signals generated by a real vehicle, with the output of the vestibular system when its input is generated by the washout filter whereby the input to the washout filter is the input signal generated by a real vehicle. The comparison then becomes a measure of the experienced difference between driving the simulator and a real car.

In this chapter the principles of motion cuing are discussed together with the tilt coordination procedure which is used in many washout filters.

## *6.1. Outputs of the Vehicle*

The vehicle model gives two outputs, acceleration $a_{SI}$ and angles $\beta_S$ which are vectors of three elements. The interval in which these variables vary can be determined by observing real car motion.

A relatively fast car accelerates from 0 to 100 km/h in 7 seconds. From here the acceleration can be found:

$$a = \frac{v - v_0}{t} = 3.97 m/s^2 \qquad (6.1)$$

Therefore in the further calculation a maximum acceleration of $a = 4 m/s^2$ will be used.

In order to find the sideward acceleration it could be assumed that a car is not taking curves with bigger speed than 90km/h in a bend with radius 100m. The centrifugal acceleration can be calculated as:

$$a_c = \frac{v^2}{R} = 6.25 m/s^2 \qquad (6.2)$$

However, several test drivers suggested that this high acceleration produces unrealistic feelings and intensive roll rotations. Moreover with normal drive, the acceleration rarely goes above $4 m/s^2$. Therefore, the maximum side acceleration was taken to be $a_c = 4 m/s^2$.
For the yaw rate it can be estimated:

$$\omega_\psi = \frac{v}{r} = 0.25 rad/s \approx 14^\circ/s \qquad (6.3)$$

In real cases the yaw angle can rotate in $360^\circ$ but the platform limitation does not allow that. The roll angle is never bigger than $\pm 10^\circ$. The pitch angle can be found from a relatively high steep hill of 30% which corresponds to $27^\circ$. Generally the rotation rate in any direction will be taken to be $\leq 0.2 rad/s$.

The real data taken from the software (implemented in Matlab) which controls the CVS of about 90s is given in Figure C.1 and Figure C.2. This data gives the acceleration and rotation a real car should have, taking the readings from the CVS.
We should be careful not to push the platform close to its limits because the low frequency translational acceleration contributes in increasing the tilt of the platform. This is described in more details in section 6.4.

## 6.2. The general framework of motion cuing

The framework of motion cuing is presented on Figure 6.1. Two input signals enter into the simulator, given with the vector:

$$u_{AI} = u_A = \begin{bmatrix} a_A \\ \beta_A \end{bmatrix} \quad (6.4)$$

These inputs are then propagated to two channels, the real car and the simulator. The signals are transformed into the driver's frame and their values at the center of the driver's head are represented with:

$$u_{EE} = \begin{bmatrix} f_{EE} \\ \omega_{EE} \end{bmatrix} \text{ (car)} \quad (6.5)$$

$$u_{DD} = \begin{bmatrix} f_{DD} \\ \omega_{DD} \end{bmatrix} \text{ ( simulator)} \quad (6.6)$$

At the end the error is estimated. This could be done in two ways, either by using the signals at the driver's head (Equation ( 6.5 ) and ( 6.6 )) both in the simulator and in a real car or by comparing the sensed signals obtained from the model of the vestibular system (Figure 6.1). The sensed signals are represented by:

$$\hat{u}_{EE} = \begin{bmatrix} \hat{f}_{EE} \\ \hat{\omega}_{EE} \end{bmatrix} \text{ (car)} \quad (6.7)$$

$$\hat{u}_{DD} = \begin{bmatrix} \hat{f}_{DD} \\ \hat{\omega}_{DD} \end{bmatrix} \text{ ( simulator)} \quad (6.8)$$

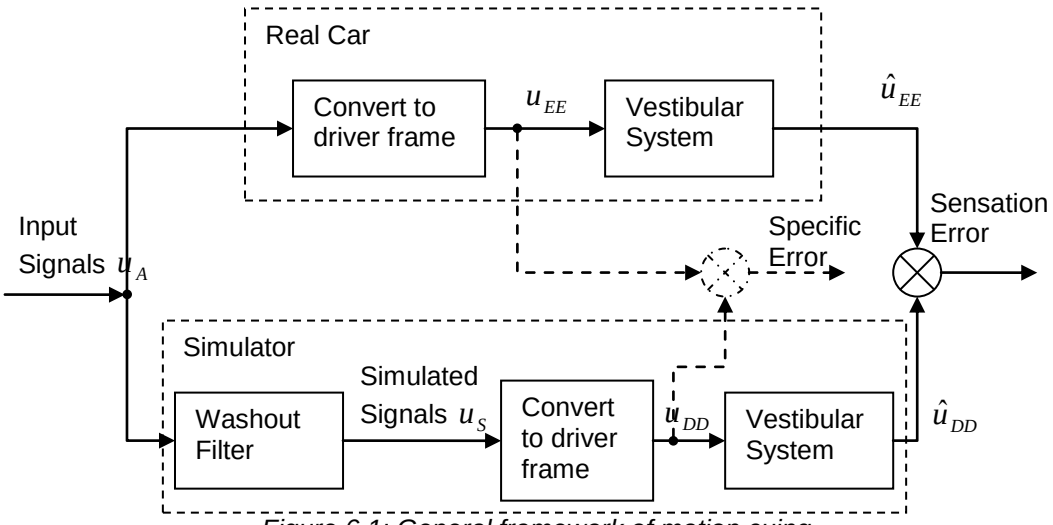


*Figure 6.1: General framework of motion cuing*

## 6.3. Specific Force and Rotation felt by the Driver

As described in chapter 4, the driver feels rotational motion (angular velocity) and specific forces. The input signals, $a_{AI}$ and $\beta_{AI}$ need to be converted to specific force $f_{EE}$ ( $f_{DD}$ ) and rotation $\omega_{EE}$ ( $\omega_{DD}$ ), and propagated to the driver's head. The conversion of the variables is done in several steps described in the data flow diagrams on Figure 6.2 and Figure 6.3.

Starting from the end, the goal is achieving similar sensed signals:

$$\hat{f}_{DD} \approx \hat{f}_{EE} \qquad (6.9)$$

$$\hat{\omega}_{DD} \approx \hat{\omega}_{EE} \left( \hat{\omega}_S \approx \hat{\omega}_A \right) \qquad (6.10)$$

According to the Equation ( 4.8 ), the specific force and rotation at the driver's head in the car can be calculated as:

$$f_{EE} = f_{EA} = a_{EA} - g_A = a_{EA} - L_{IA} g_I \qquad (6.11)$$

Because frame *E* is parallel to frame *A* the vector of gravity is same in these frames:

$$g_E = g_A \qquad (6.12)$$

The acceleration $a_{EA}$ can be computed as:

$$a_{EA} = a_{AA} + A_{EA} r_{EA} \qquad (6.13)$$

where the matrix $A_{EA}$ is described in equation ( 3.21 ) and the vector $r_{EA}$ represents the position of the driver's head in the car in respect to frame *A*. The value of $r_{EA}$ in this research is:

$$r_{EA} = \begin{bmatrix} 0 \\ 0 \\ -1.3 \end{bmatrix} \qquad (6.14)$$

According to Equation ( 3.21 ), the matrix $A_{EA}$ implies evaluation of the derivatives $\dot{\beta}_A$ and $\dot{\omega}_A$ , where $\omega_A$ according to Equation ( 3.12 ) is calculated as:

$$\omega_A = R_A \dot{\beta}_A \qquad (6.15)$$

and the matrix $R_A$ was given in Equation ( 3.14 ).

The specific force and rotation at the driver's head in the simulator is calculated in the similar way, the only difference is the washout filter, added due to the platform limitation. The equations for the simulator are:

$$f_{DD} = f_{DA} = a_{DS} - g_S = a_{DS} - L_{IS} g_I \qquad (6.16)$$

$$a_{DS} = a_{SS} + A_{DS} r_{DS} \qquad (6.17)$$

$$\omega_S = R_S \dot{\beta}_S \qquad (6.18)$$

There are several relations in the above equations:

$$L_{IS} = L_{IA} \quad (6.19)$$

$$A_{DS} = A_{EA} \quad (6.20)$$

$$R_S = R_A \quad (6.21)$$

$$r_{DS} = r_{EA} \quad (6.22)$$

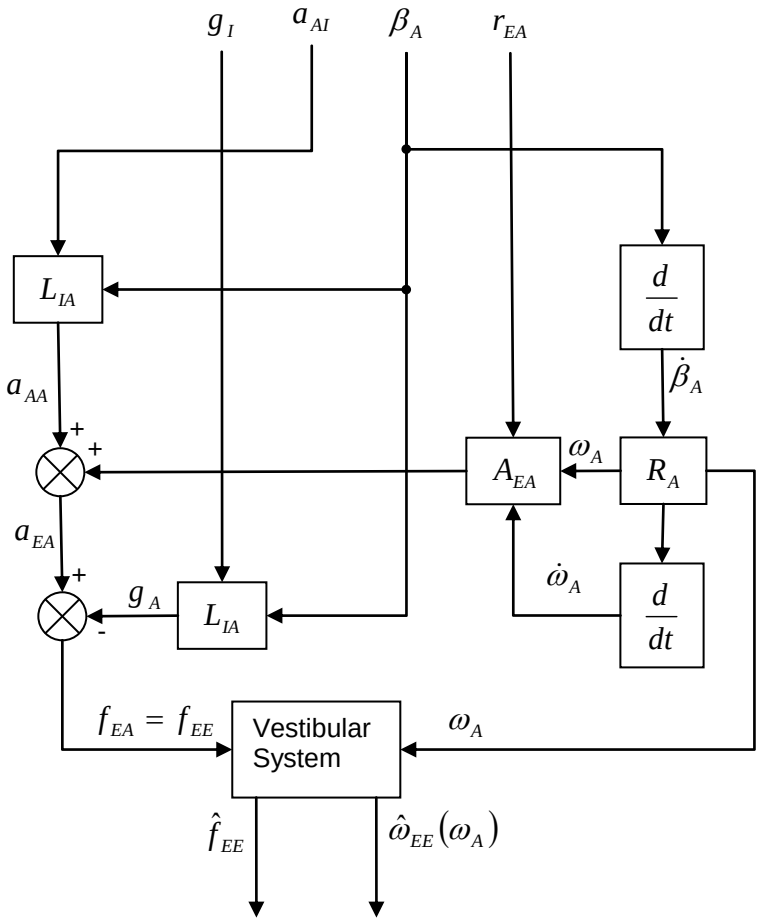


*Figure 6.2: Calculation of specific ( $f_{EA}, \omega_A$) and sensed ( $\hat{f}_{EE}, \hat{\omega}_{EE}$) forces and rotation in a car*

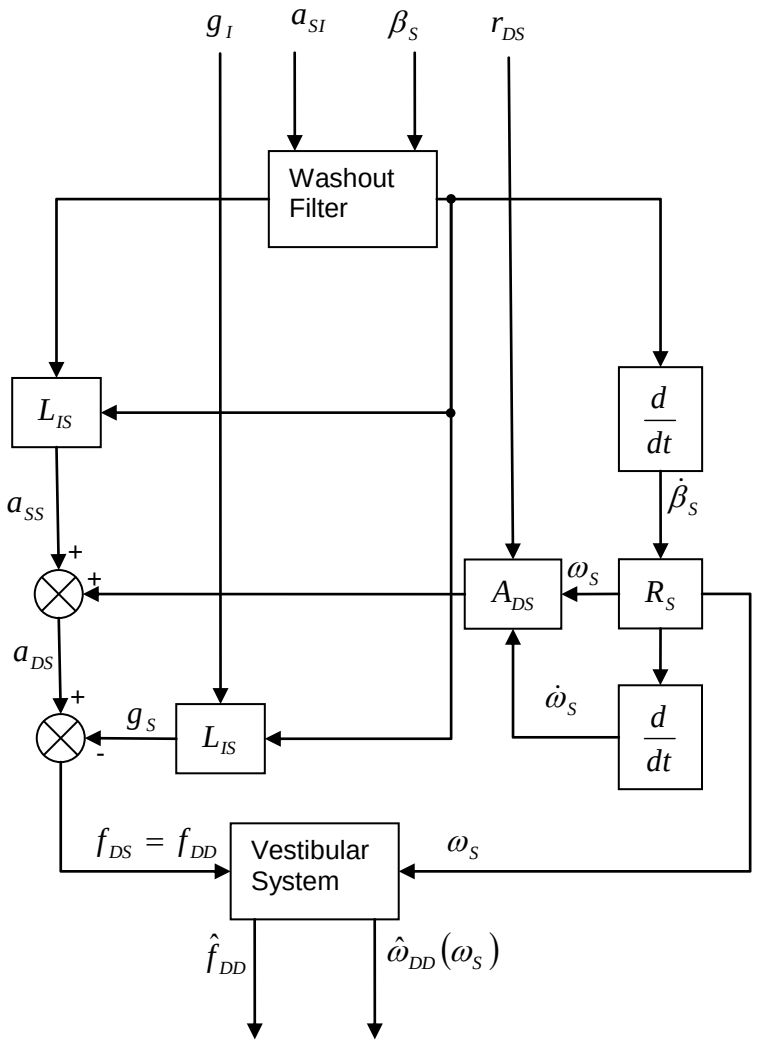


*Figure 6.3: Calculation of specific ( $f_{DS}, \omega_S$ ) and sensed ( $\hat{f}_{DD}, \hat{\omega}_{DD}$ ) forces and rotation in the simulator*

## 6.4. Tilt Coordination

The sustained translational acceleration is sensed by the driver as a long term change in the magnitude and direction of the specific force. Generally, this can not be simulated due to the limits of the platform which can barely move 25cm. Yet, it is possible to reproduce the same feeling by tilting the platform so that the driver feels the force of gravity as translational acceleration. Tilting the platform should be done with small rate of change of the Euler angles so that the driver does not feel rotation.

This transformation from acceleration to Euler angles can be done for movements in *x* and *y* direction and the magnitude is limited to the gravitational constant *g*. Accelerations in *x* direction can be simulated by tilting the platform with a pitch angle $\theta_S$ around the *y*-axis (Figure 6.4). The gravity vector can be decomposed into its *x* and *z* component, where the *x* component simulates the translational acceleration $a_{SS_x}$ opposite to the gravity.

$$g = -a_{SS} = \begin{bmatrix} -g\sin(\theta_S) \\ 0 \\ g\cos(\theta_S) \end{bmatrix} \quad (6.23)$$

The acceleration in *y* direction can be represented by tilting the platform with roll angle $\phi_S$:

$$g = -a_{SS} = \begin{bmatrix} 0 \\ g\sin(\phi_S) \\ -g\cos(\phi_S) \end{bmatrix} \quad (6.24)$$

The general equation for the simulated translational acceleration becomes:

$$a_{SS} = \begin{bmatrix} g\sin(\theta_S) \\ -g\cos(\theta_S)\sin(\phi_S) \\ -g\cos(\theta_S)\cos(\phi_S) \end{bmatrix} \quad (6.25)$$

Because the angles are smaller than $20^\circ$ we can substitute $\sin(\alpha) \approx \alpha, \cos(\alpha) \approx 1$:

$$a_{SS} = \begin{bmatrix} g\theta_S \\ -g\phi_S \\ -g \end{bmatrix} \quad (6.26)$$

From here the Euler angles can be expressed as:

$$\beta_S = \begin{bmatrix} 0 & -\frac{1}{g} & 0 \\ \frac{1}{g} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} a_{SS} = TCa_{SS},\ TC = \begin{bmatrix} 0 & -\frac{1}{g} & 0 \\ \frac{1}{g} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (6.27)$$

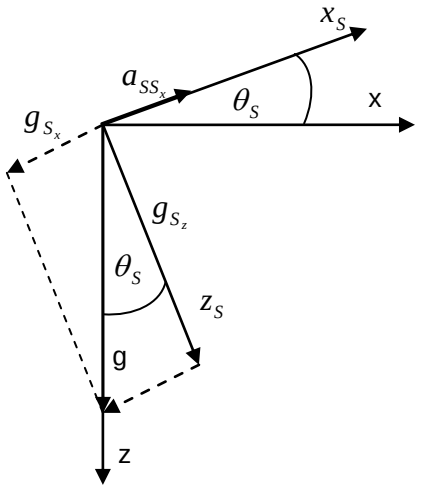


*Figure 6.4: Simulated acceleration about the x-axis*

## *6.5. Washout Filter Alternatives*

Several implementations of washout filters exist in practise ([ 1 ], [ 7 ], [ 8 ], [ 9 ] and [ 10 ]). The most common is the classical washout filter which is one of the oldest versions. More complex versions are the optimal, adaptive and nonlinear washout filter.

The Optimal WF contains a model of the vestibular system and the Adaptive WF uses feedback to tune its parameters depending on the platform state. The nonlinear WF uses the benefits from the optimal and the adaptive WF by implementing vestibular system and using feedback for tuning the parameters in real time. These filters are the most promising and this research will be focused mostly on them. There is also a max-tilt washout filter (Section 8) which is a modified version of the Classical WF. The idea behind this filter is using as much tilt as possible on the motion platform. Because this filter is very similar to the Classical WF it is not investigated in this work and only its basic description is given.

The washout filters are optimized by the use of optimization algorithms described in Appendix D.

# 7. Classical Washout Filter (CWF)

The basic idea of the CWF is creating specific forces and rotations at the driver's head in the simulator similar to those they would experience in a real car [ 7 ]. This was more described in section 6.3. In this research the quality of the filter is estimated by minimizing the sensation error, i.e. the difference between the sensed specific forces and rotations from the simulator and a real car. The CWF does not include the vestibular system inside, though it is used during the optimization.

The CWF consists of three channels, translational, rotational and coordination channel, Figure 7.1.

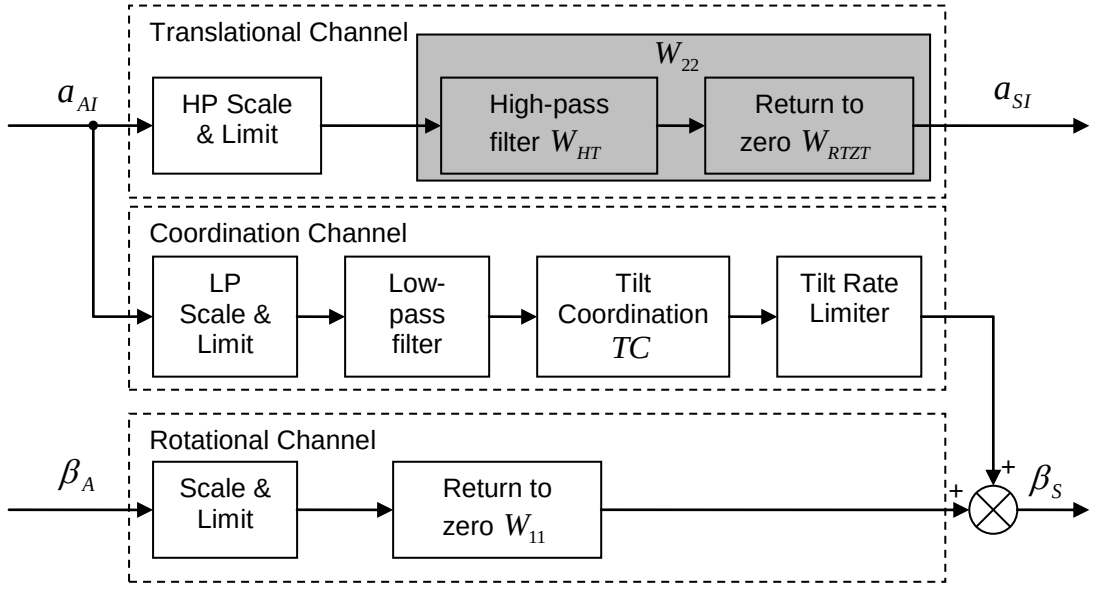


*Figure 7.1: Classical washout filter*

## *7.1. Translational Channel*

The translational channel treats the translational acceleration of the frame *A* in respect to the inertial frame, $a_{AI}$. The acceleration is first scaled and limited in order to restrict the motion based travel to remain within the physical system constraints (during the optimization of the CWF, the scale coefficients had value of one, and the platform was optimized by using relatively small inputs. After the optimization these values are adjusted again so that the platform can work close to the limits but never to exceed them). The scale and limit block is described in more details in section 7.4.

Because the platform cannot reproduce much of the lower frequencies of $a_{AI}$, the acceleration is then filtered through a high-pass filter (the distance is a second integral of the acceleration, for example a constant acceleration $a_c$ will lead to $s = a_c \frac{t^2}{2}$ which increases rapidly for a very short time). For this purpose a second order filter is used, with transfer function:

$$W_{HT} = \frac{s^2}{s^2 + 2\zeta_t \omega_t s + \omega_t^2} \qquad (7.1)$$

where $\varsigma_t$ is the relative damping coefficient and $\omega_t[rad/s]$ is called the natural frequency.

When the input acceleration has constant value for a while, the platform must slowly return to the neutral position in order to have good starting point for the future accelerations. For this reason another high pass filter is added, return to zero high-pass filter. The transfer function for this filter is:

$$W_{RTZT} = \frac{s}{s+\omega_b} \qquad (7.2)$$

After these two filters the acceleration will eventually return to zero. The parameters for the filters are computed for each direction separately. Joining them together the final filter for the translational canal is produced, giving:

$$W_{22} = W_{HT} W_{RTZ} \qquad (7.3)$$

## 7.2. Coordination Channel

As was discussed in section 6.4 the sustained translational acceleration is sensed by the driver as a long term change in the magnitude and direction of the specific force. In order to simulate this motion the tilt coordination methods were used so that the driver feels the force of gravity as translational acceleration. Therefore the coordination channel is needed to transform the acceleration $a_{AI}$ into Euler angles. This transformation from acceleration to Euler angles can be done only for movements in *x* and *y* direction.

First the input acceleration is scaled and limited and then it is passed trough a low pass filter to sort out the sustained acceleration. A second order filter is used with transfer function:

$$W_{12} = \frac{\omega_c^2}{s^2 + 2\varsigma_c \omega_c s + \omega_c^2} \qquad (7.4)$$

The resulting signal is then converted into Euler angles by passing through the tilt coordination matrix $TC$ given in Equation ( 6.27 ).

One should be careful to not produce very big tilt rate. The driver should not feel rotation during this process. Therefore in this channel, a tilt rate limiter is used. The driver starts to feel rotation on rate bigger than $0.2rad/s$, so the coordination channel is prevented to produce higher tilt rate.

## 7.3. Rotational Channel

The rotational channel is directly responsible for the rotational movement of the platform. The Euler angles $\beta_A$ come as one input and the second input comes from the coordination channel. The first component intends to simulate the rotational velocity of the simulator and the second component is intended to tilt the simulator in order to simulate sustained accelerations.

The signal $\beta_A$ is first scaled and limited, and then is passed through a first order high-pass filter which can be interpreted as return to zero filter:

$$W_{11} = \frac{s}{s + \omega_r} \qquad (7.5)$$

After this step the final output angle $\beta_S$ is calculated as a sum of the filtered $\beta_A$ and the tilt angle coming from the coordination channel.

## 7.4. Scale and Limit Block

The scaling and limiting block is used to reduce the motion response of the platform. A linear scaling is used in order to modify the amplitude uniformly across all frequencies. On the other hand, the limiting is a nonlinear process. It is used to prevent the signal rise above a preselected magnitude. There are three such blocks in the CWF.

One should be careful to not scale the acceleration with the gravitational component within. If the acceleration is presented as:

$$a = a_1 + g \qquad (7.6)$$

Then the acceleration after the scaling and limiting block should be:

$$a_{scaled} = Ka_1 + g \qquad (7.7)$$

The scaling matrix must be diagonal (with other words, each direction is scaled separately) in order to avoid cross-coupling among the components in $a_{AA}$ and $a_{SS}$ [ 7 ].

The input/output characteristics of this block are shown on Figure E.1.

## 7.5. High-pass and Low-pass Filters

The high-pass filter is used to block the undesired low frequencies, and the low-pas filter is used to block the undesired high frequencies.

High-pass filters are especially needed because the low frequencies tend to extend the actuator lengths. These filters are well described by Reid and Nahon [ 7 ] and can be represented with transfer function of form:

$$W = \frac{s^n N(s)}{D(s)} \qquad (7.8)$$

where

$$N(s) = \sum_{i=0}^{k} a_i s^i \qquad (7.9)$$

$$D(s) = \sum_{j=0}^{m} b_j s^j \qquad (7.10)$$

$$m \geq n + k \qquad (7.11)$$

$$a_0, b_0 \neq 0 \qquad (7.12)$$

The design of the high-pass filter will be found by using constant acceleration as input signal:

$$a_A = A \qquad (7.13)$$

The displacement of the actuators, taking they were in zero position before, and represented in Laplace form, can be calculated as:

$$\tilde{x}_S = \frac{\tilde{a}_{Afiltered}}{s^2} = \tilde{a}_A s^{n-2} \frac{N(s)}{D(s)} \qquad (7.14)$$

where $\tilde{a}_A$ is Laplace transformation of the input acceleration and can be written as:

$$\tilde{a}_A = \frac{A}{s} \qquad (7.15)$$

Finally for the displacement of the platform we get:

$$\tilde{x}_S = A s^{n-3} \frac{N(s)}{D(s)} \qquad (7.16)$$

$$x_S(t) = A s^{n-2} \frac{N(s)}{D(s)} \qquad (7.17)$$

The simulator should return in its neutral position after some time, even if the input acceleration has the same constant value.

$$\lim_{t\to\infty} x_S(t) = \lim_{s\to 0} A s^{n-2} \frac{N(s)}{D(s)} \qquad (7.18)$$

From here it follows that the minimum value for *n* should be:

$$n \geq 3 \qquad (7.19)$$

The transfer function of the translational filter together with the return to zero filter for minimum value of *n* will look like:

$$W_{HT} W_{RTZT} = \frac{s^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \frac{s}{s + \omega_b} \qquad (7.20)$$

In the similar way, the minimum value of *n* can be found for the high-pass filter in the rotation channel:

$$n \geq 1 \qquad (7.21)$$

with transfer function:

$$W_{HR} = W_{RTZR} = \frac{s}{s + \omega_n} \qquad (7.22)$$

## 7.6. CWF optimization

The classical washout filter is optimized with evolutionary algorithms. One could find the CWF parameters by hand, but of course EA will give better solution. First the parameters are manually adjusted and this was the starting point for EA. The goal of EA is to adjust the parameters such that the sensed force and rotation one should feel in the simulator is as close as possible to the sensation one should feel in a real car, Section 6.2.

EA are population based algorithms (Appendix D.1) and for each individual the simulation is run for about 10 to 14 seconds and then the fitness is calculated as an inverse function of the error between the sensation curves from the car and the simulator. The simulation is done in Matlab, Simulink, and the model is given in Figure E.11.

There is one constraint that should not be broken during the whole simulation, that is, the actuators should not exceed over their limits. The fitness of the individuals must be adjusted so that individuals who kept the platform within its boundaries during the whole simulation should always have bigger fitness than the others who exceeded the limits. Therefore the fitness of the individuals is computed as:

$$fitness = \begin{cases} \dfrac{t}{T_{\max}} & for\ t < T_{\max} \\ 1 + \dfrac{1}{err} & otherwise \end{cases} \qquad (7.23)$$

where $T_{\max}$ is the maximum simulation time, predefined from the user; *t* is the actual simulation time taking that the simulation can end before reaching $T_{\max}$ if the actuator lengths exceeded; *err* is the error between the sensation curves from the car and the simulator and is calculated as:

$$err = \sum_{t=0}^{T_{\max}} \left( x_s(t) - x_c(t) \right)^2 \qquad (7.24)$$

where $x_s$ is the sensed signal from the simulator and $x_c$ is the sensed signal from the car.

There is in total 16 parameters that need to be tuned, Table E.2. However, not all parameters are optimized at the same time. It was found that EA can be run in four separate modes. These modes are pitch/surge, roll/sway, yaw and heave.

### 7.6.1. Pitch/Surge mode

Let us consider surge acceleration as input and zero value for all other inputs. Moreover we can take that the input is a step signal (sudden acceleration). We expect the platform move quickly in longitudinal direction to catch the rising edge and then tilt with some pitch angle to simulate the sustained acceleration using the tilt coordination method described in section 6.4. Because there is no other actual pitch angle we can assume that the pitch value generated as a result of tilt coordination is propagated unmodified as a pitch input in the next measurement. This is represented as a feedback connection from the output to the input pitch of the CWF, Figure E.11.

In this way, while optimizing the parameters for the surge acceleration as input, we need to optimize the parameters for the pitch angular displacement as well. There are 6 values that need to be optimized in this mode, as can be seen from Table E.3.

CWF is optimized by taking sudden acceleration and sudden brake as an input signal. Then it is tested on filtered white noise and chirp signal as surge acceleration (Figure E.6). It was found that the platform can successfully simulate acceleration in the interval $[-2,2]m/s^2$. It might simulate slightly bigger accelerations, but this is not of very big interest, because in real motion the input will be rescaled to even smaller values. The problem occurs when the driver joins two manoeuvres at the same time, for example sudden surge acceleration and harsh lane change. The actuators' lengths can easily exceed in this case. Therefore, the optimization is done with scaling 1 (Table E.5) and signals limited in [-2, 2] and after the optimization the real input signals (the surge acceleration can go up to $4m/s^2$) are rescaled to values smaller than [-2, 2]. The final values for scale and limit are given in Table E.6.

### 7.6.2. Roll/Sway mode

The Roll/Sway mode works similarly like the Pitch/Surge mode. When sway acceleration is present as input, the platform should move in lateral direction supported by tilt coordination with roll angle. In the same way it can be assumed that there is a feedback connection from the output to the input roll of the CWF, Figure E.11.

As an input, periodic lane change manoeuvre is used and after the optimization the filter is tested with filtered white noise and chirp signal as sway acceleration (Figure E.7). During the optimization the scaling on the input signal is 1 and it is limited in the interval $[-2,2]m/s^2$.

There are 6 parameters that need to be optimized in this mode (Table E.3).

### 7.6.3. Yaw mode

The yaw mode optimizes angular displacement in yaw direction, $\psi$. A periodic lane change manoeuvre is used as an input signal. Later it is tested with filtered white noise and chirp signal (Figure E.8). During the optimization the scaling is 1 and the signal is limited in the interval $[-0.2,0.2]rad$.

There is only one parameter that needs to be optimized in this mode (Table E.3).

### 7.6.4. Heave mode

Low frequency motion in this direction rarely happens, besides that, only high frequency acceleration can be simulated. Tilt coordination can not be applied in this mode and this is the main reason way the sustained heave acceleration is omitted.

As in input signal a motion is taken such that the vehicle goes to a sudden downhill slop and then to a sudden uphill slop. There are 3 parameters that need to be optimized in this mode (Table E.3).

# 8. Max-Tilt Washout Filter (MTWF)

This section is based on MTWF representation from the work of Pieter van Balen [ 6 ]. The MTWF is a modified version of the CWF. The main idea of this filter is using as much tilt as possible on the motion platform. Its implementation is shown on Figure 8.1. The change is made in the coordination channel, the low-pass filter is removed and only the Tilt coordination block remains with the tilt rate limiting within.

The result of this filter is strong feel of acceleration on the driver. The gravity components used to simulate the sustained accelerations are subtracted from the original acceleration. The further development is very similar to the development used in CWF.

During the test drives on the simulator it was found that very large tilt causes unrealistic movement. This is partly of the fact that the present graphical environment cannot deal very well with large tilt angles. For example when the driver accelerates with high ramp slope, the simulator causes very big tilt in pitch direction and the graphical environment also tilts the car in respect to the road making the driver look in the sky.

This WF is not developed in this paper but could be a warm up exercise for future developers.

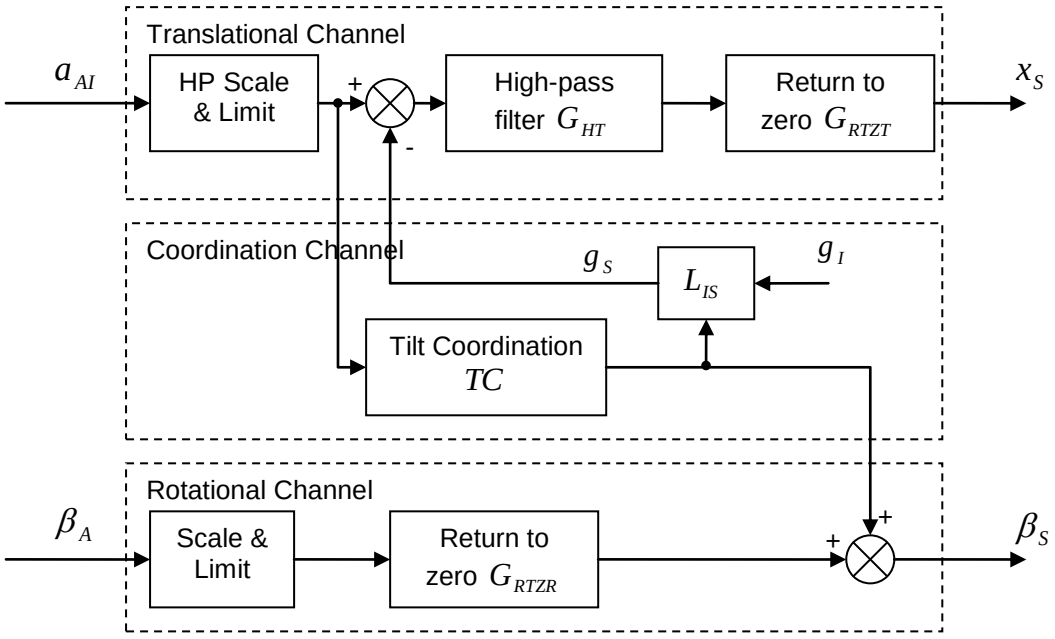


*Figure 8.1: Max-Tilt washout filter [ 6 ]*

# 9. Optimal Washout Filter (OWF)

The development of OWF is based on the work of Telban and Cardullo [ 1 ] and Reid and Nahon [ 7 ] and [ 8 ]. The difference between OWF and CWF is that the system of equations characteristic for OWF have the vestibular system included inside, thus producing higher order filters. The quality of OWF is determined by minimizing the sensation error between the sensed specific forces and rotations produced by the model of the vestibular system in the simulator and the sensed specific forces and rotations produced by the model of the vestibular system in the real car. The process of error estimation was shown on Figure 6.1.

The desired transfer function of the high and low pass filters is determined offline by using the mathematical model of the vestibular system. Then, after the form and the parameters of the filters are determined, OWF is implemented online, on the Chalmers Vehicle Simulator.

The OWF consist of four filters, $W_{11}, W_{12}, W_{21}, W_{22}$ and its implementation is shown on Figure 9.1 (it was found later that $W_{21}$ does not contribute in the final solution and therefore it was removed. This is better described in Chapter 13 - Discussion).

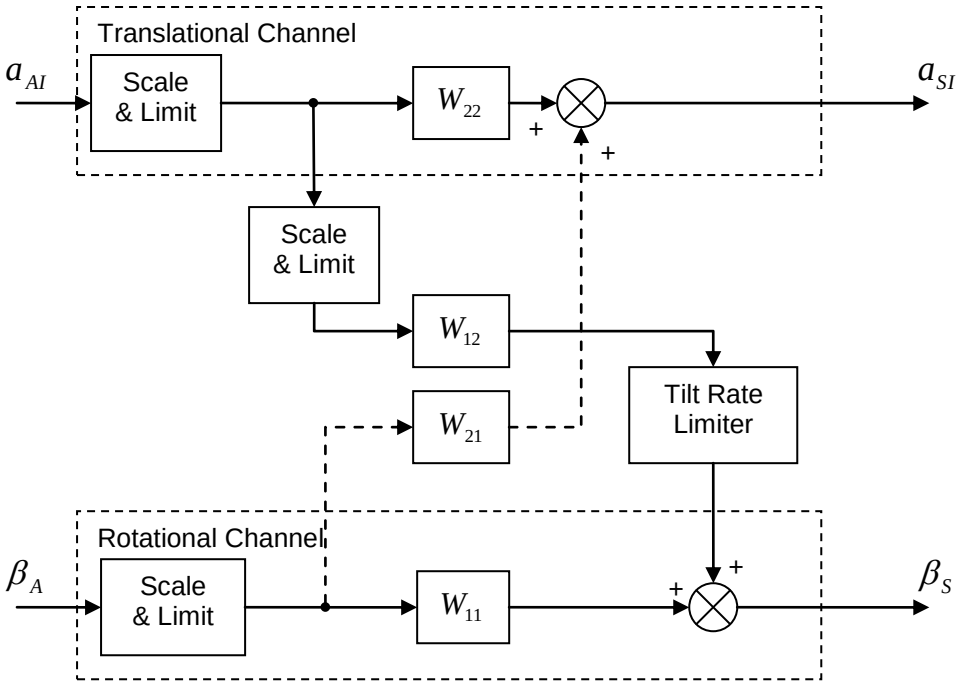


*Figure 9.1: Optimal washout filter*

The problem is to determine the transfer function $W(s)$ (including the all four components) such that the estimated error between the sensed signals of the car and the simulator is minimized (Figure 9.2). The input values in the car are represented by $\underline{u}_a$, and by $\underline{u}_S$ in the simulator, and their relationship including the transfer function is:

$$\underline{u}_S = \underline{W}(s)\underline{u}_a,\ \underline{W}(s) = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} \qquad (9.1)$$

The motion sensation in the car is $\underline{y}_a$, and $\underline{y}_S$ in the simulator. One need to represent additional motion variables, symbolized as $\underline{y}_{d_a}$ in the car, and $\underline{y}_{d_S}$ (Equation ( F.24 ), ( F.43 ), ( F.61 ) and ( F.79 )) in the simulator which will help in finding optimal solution.

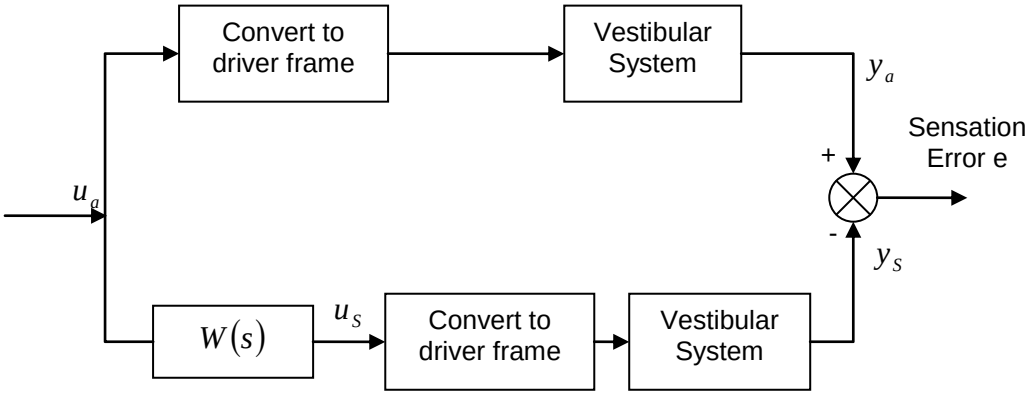


*Figure 9.2: Development of OWF*

## 9.1. Problem structure and dynamics

By using mathematical model of the vestibular system (Section 4.2), the desired function $W(s)$ is generated offline, in a computer simulation, and then implemented online on the CVS.

One needs to choose in which reference frame OWF should be applied. Telban and Cardullo [ 1 ] used the simulator reference frame with the center of rotation located at the centroid of the simulator motion base. Reid and Nahon [ 7 ], [ 10 ] used the frame at the pilot's head. Both choices have some advantages and disadvantages. The Reid and Nahon's method eliminates sensation cross-coupling, but makes the development of $W(s)$ more complicated.
In this work the WF is applied at the driver's head.

Next, one needs to decide which control inputs are most appropriate. Reid and Nahon used longitudinal and lateral linear acceleration and angular displacement as control input. Telban and Cardullo used translational accelerations, but angular velocity instead of displacement. Since during the development of CWF the angular displacement was used as one of the inputs, it was used in the development of OWF too. Thus the input vector has the form:

$$u = \begin{bmatrix} \underline{\beta} \\ \underline{a} \end{bmatrix} \qquad (9.2)$$

The dynamics of the motion sensing system are formulated as linear differential equations both in the car (the upper branch on Figure 9.2) and in the simulator (the lower branch on Figure 9.2). It is assumed that the same sensation model (model of vestibular system) is applied to the simulator and the car and all system matrices are taken to be time invariant:

$$\dot{\underline{x}}_a = \underline{A}_v \underline{x}_a + \underline{B}_v \underline{u}_a \qquad (9.3)$$

$$\underline{y}_a = \underline{C}_v \underline{x}_a + \underline{D}_v \underline{u}_a \qquad (9.4)$$

$$\dot{\underline{x}}_S = \underline{A}_v \underline{x}_S + \underline{B}_v \underline{u}_S \qquad (9.5)$$

$$\underline{y}_S = \underline{C}_v \underline{x}_S + \underline{D}_v \underline{u}_S \qquad (9.6)$$

In order to cover many input scenarios, filtered white noise is used as input which can be represented with the state space model:

$$\dot{\underline{x}}_n = \underline{A}_n \underline{x}_n + \underline{B}_n \underline{u}_n \quad (9.7)$$

$$\underline{u}_a = \underline{x}_n \quad (9.8)$$

$$\underline{A}_n = \begin{bmatrix} -\gamma_1 & 0 \\ 0 & -\gamma_2 \end{bmatrix},\ \underline{B}_n = \begin{bmatrix} \gamma_1 \\ \gamma_2 \end{bmatrix} \quad (9.9)$$

where $\gamma_1$ and $\gamma_2$ are the first-order filter break frequencies for each degree of freedom.
The additional internal variables (given in Equation ( 9.65 ) and Table F.1) can be expressed as:

$$\dot{\underline{x}}_d = \underline{A}_d \underline{x}_d + \underline{B}_d \underline{u}_S \quad (9.10)$$

The vestibular system model, containing both the otolith and the semicircular canal, in state space representation can be written as:

$$\dot{\underline{x}}_v = \underline{A}_v \underline{x}_v + \underline{B}_v \underline{u} \quad (9.11)$$

$$\underline{y}_v = \underline{C}_v \underline{x}_v + \underline{D}_v \underline{u} \quad (9.12)$$

where $\underline{u}$ and $\underline{y}_v$ stands both for the car and the simulator.

The pilot sensation error $\underline{e} = \underline{y}_S - \underline{y}_a$ and the vestibular state error $\underline{x}_e = \underline{x}_S - \underline{x}_a$ are:

$$\dot{\underline{x}}_e = \underline{A}_v \underline{x}_e + \underline{B}_v \underline{u}_S - \underline{B}_v \underline{u}_a \quad (9.13)$$

$$\underline{e} = \underline{C}_v \underline{x}_e + \underline{D}_v \underline{u}_S - \underline{D}_v \underline{u}_a \quad (9.14)$$

Equations ( 9.7 ) - ( 9.14 ) can be collected together into one state space system:

$$\underline{x} = \begin{bmatrix} \underline{x}_e^T & \underline{x}_d^T & \underline{x}_n^T \end{bmatrix}^T,\ \underline{y} = \begin{bmatrix} \underline{e}^T & \underline{x}_d^T \end{bmatrix}^T \quad (9.15)$$

$$\dot{\underline{x}} = \underline{A}\underline{x} + \underline{B}\underline{u}_S + \underline{H}\underline{u}_n \quad (9.16)$$

$$\underline{y} = \underline{C}\underline{x} + \underline{D}\underline{u}_S \quad (9.17)$$

where:

$$\underline{A} = \begin{bmatrix} \underline{A}_v & \underline{0} & -\underline{B}_v \\ \underline{0} & \underline{A}_d & \underline{0} \\ \underline{0} & \underline{0} & \underline{A}_n \end{bmatrix},\ \underline{B} = \begin{bmatrix} \underline{B}_v \\ \underline{B}_d \\ \underline{0} \end{bmatrix},\ \underline{H} = \begin{bmatrix} \underline{0} \\ \underline{0} \\ \underline{B}_n \end{bmatrix} \quad (9.18)$$

$$\underline{C} = \begin{bmatrix} \underline{C}_v & \underline{0} & -\underline{D}_v \\ \underline{0} & \underline{I} & \underline{0} \end{bmatrix},\ \underline{D} = \begin{bmatrix} \underline{D}_v \\ \underline{0} \end{bmatrix} \quad (9.19)$$

The cost function used for minimizing the error is of form:

$$J = E\left\{\int_{t_0}^{t_1} \left[\underline{e}^T \underline{Q}\underline{e} + \rho\left(\underline{u}_S^T \underline{R}\underline{u}_S + \underline{x}_d^T \underline{R}_d \underline{x}_d\right)\right] dt\right\} \quad (9.20)$$

where $\underline{Q}$, $\underline{R}_d$ and $R$ are positive symmetric matrices and $\rho$ is a positive scalar. The cost function, as shown in Appendix F, takes the form:

$$J = E\left\{\underline{x}^T \underline{R}'\underline{x} + \rho \underline{u}'^T_S \underline{R}_2 \underline{u}'\right\} \quad (9.21)$$

$$\dot{\underline{x}} = \underline{A}'\underline{x} + \underline{B}\underline{u}' + \underline{H}\underline{u}_n \quad (9.22)$$

Collecting all other parameters derived in Appendix F.1, we have:

$$\underline{A}' = \underline{A} - \underline{B}\underline{R}_2^{-1}\underline{R}_{12}^T,\ \underline{u}' = \underline{u}_S + \underline{R}_2^{-1}\underline{R}_{12}^T\underline{x},\ \underline{R}'_1 = \underline{R}_1 - \underline{R}_{12}\underline{R}_2^{-1}\underline{R}_{12}^T,$$
$$\underline{R}_1 = \underline{C}^T\underline{G}\underline{C},\ \underline{R}_{12} = \underline{C}^T\underline{G}\underline{D},\ \underline{R}_2 = \rho\underline{R} + \underline{D}^T\underline{G}\underline{D},\ \underline{G} = \begin{bmatrix} \underline{Q} & \underline{0} \\ \underline{0} & \rho\underline{R}_d \end{bmatrix} \quad (9.23)$$

and the cost function is minimized when:

$$\underline{u}' = -\underline{R}_2^{-1}\underline{B}^T\underline{P}\underline{x} \quad (9.24)$$

Where the matrix $\underline{P}$, according to the equations in Appendix F.1, is the solution to the algebraic Riccati equation:

$$\underline{0} = \underline{R}_1^{'} - \underline{P}\underline{B}\underline{R}_2^{-1}\underline{B}^T\underline{P} + \underline{A}^{'T}\underline{P} + \underline{P}\underline{A}^{'} \quad (9.25)$$

From Equation ( 9.23 ) and ( 9.24 ) one can obtain:

$$\underline{u}_S = -\underline{R}_2^{-1}\left(\underline{B}^T\underline{P} + \underline{R}_{12}^T\right)\underline{x} = -\underline{F}\underline{x} \quad (9.26)$$

The matrix $\underline{F}$ can be partitioned in the following way:

$$\underline{u}_S = -\begin{bmatrix}\underline{F}_1 & \underline{F}_2 & \underline{F}_3\end{bmatrix}\begin{bmatrix} \underline{x}_e^T \\ \underline{x}_d^T \\ \underline{x}_n^T \end{bmatrix} \quad (9.27)$$

Now collecting Equations ( 9.8 ), ( 9.18 ), ( 9.19 ) and ( 9.22 ), and neglecting the $\underline{x}_n$ states, one can obtain:

$$\begin{bmatrix} \dot{\underline{x}}_e^T \\ \dot{\underline{x}}_d^T \end{bmatrix} = \begin{bmatrix} \underline{A}_v & \underline{0} & -\underline{B}_v \\ \underline{0} & \underline{A}_d & \underline{0} \end{bmatrix}\begin{bmatrix} \underline{x}_e^T \\ \underline{x}_d^T \\ \underline{u}_a^T \end{bmatrix} + \begin{bmatrix} \underline{B}_v \\ \underline{B}_d \end{bmatrix}\underline{u}_S \quad (9.28)$$

Equation ( 9.27 ) can be substituted into Equation ( 9.28 ) resulting in:

$$\begin{bmatrix} \dot{\underline{x}}_e^T \\ \dot{\underline{x}}_d^T \end{bmatrix} = \begin{bmatrix} \underline{A}_v - \underline{B}_v\underline{F}_1 & -\underline{B}_v\underline{F}_2 \\ -\underline{B}_d\underline{F}_1 & \underline{A}_d - \underline{B}_d\underline{F}_2 \end{bmatrix}\begin{bmatrix} \underline{x}_e^T \\ \underline{x}_d^T \end{bmatrix} + \begin{bmatrix} -\underline{B}_v\left(\underline{I} + \underline{F}_3\right) \\ -\underline{B}_d\underline{F}_3 \end{bmatrix}\underline{u}_a \quad (9.29)$$

Equation ( 9.29 ) will be substituted into Equation ( 9.27 ) in order to obtain the final transfer function:

$$\underline{W}(s) = \begin{bmatrix}\underline{F}_1 & \underline{F}_2\end{bmatrix}\begin{bmatrix} s\underline{I} - \underline{A}_v + \underline{B}_v\underline{F}_1 & \underline{B}_v\underline{F}_2 \\ \underline{B}_d\underline{F}_1 & s\underline{I} - \underline{A}_d + \underline{B}_d\underline{F}_2 \end{bmatrix}^{-1}\begin{bmatrix} \underline{B}_v\left(\underline{I} + \underline{F}_3\right) \\ \underline{B}_d\underline{F}_3 \end{bmatrix} - \underline{F}_3 \quad (9.30)$$

## 9.2. Implementation of the Vestibular Model

The control inputs in the car and the simulator are defined as:

$$\underline{u}_a = \begin{bmatrix} \underline{\beta}_a^T & \underline{a}_{AI}^T \end{bmatrix}^T,\ \underline{u}_S = \begin{bmatrix} \underline{\beta}_S^T & \underline{a}_{SI}^T \end{bmatrix}^T \qquad (9.31)$$

These inputs are applied on the vestibular system which produces the outputs. The outputs are the sensed specific force and rotation from the car and the simulator at some point *P* i.e. at the driver's head.

$$\underline{y}_a = \begin{bmatrix} \underline{\hat{\omega}}_{Pa} & \underline{\hat{f}}_{Pa} \end{bmatrix},\ \underline{y}_S = \begin{bmatrix} \underline{\hat{\omega}}_{PS} & \underline{\hat{f}}_{PS} \end{bmatrix} \qquad (9.32)$$

The linearized models for the semicircular canals and otoliths are used by deleting the threshold values, given in Equation ( 4.2 ) and ( 4.9 ).

Because the vestibular system expects rotation as one of the inputs, the relationship between the angular displacement and angular velocity needs to be calculated. According to Equation ( 3.15 ) and ( 3.16 ) and taking into account linearization about $\underline{\omega}_{SS} = \underline{0}$ and $\underline{\beta}_S = \underline{0}$ one could write:

$$\dot{\beta}_S = T_S \omega_{SS} \qquad (9.33)$$

$$T_S = \begin{bmatrix} 1 & 0 & \theta_S \\ 0 & 1 & -\phi_S \\ 0 & \phi_S & 1 \end{bmatrix} \qquad (9.34)$$

and from here by retaining only the first order terms:

$$\dot{\beta}_S = \begin{bmatrix} p_{SS} + \theta_S r_{SS} \\ q_{SS} - \phi_S r_{SS} \\ \phi_S q_{SS} + r_{SS} \end{bmatrix} \approx \begin{bmatrix} p_{SS} \\ q_{SS} \\ r_{SS} \end{bmatrix} = \omega_{SS} \qquad (9.35)$$

By observing the equation above, it can be noted that the generation of OWF does not depend on the position of the driver's head. The linearization used in the pervious equations simplifies the solution, but it could lead to not fully optimized filter. However the final result showed that this solution is satisfactory.
The Equation ( 4.2 ) for the semicircular canals now can be written as:

$$\frac{\hat{\omega}}{\beta} = \frac{T_L T_a s^3}{(T_a s + 1)(T_L s + 1)(T_4 s + 1)} \qquad (9.36)$$

where instead of $T_S$ which is used in Equation ( 4.2 ), $T_4$ stands, thus preventing any conflict with the transformation matrix $T_S$ given in Equation ( 9.34 ). The sensed rotation $\hat{\omega}$ could be replaced by each coordinate acting at the point *P* ( $\hat{p}_{PS}, \hat{q}_{PS}, \hat{r}_{PS}$ ) and for $\beta$ each angular coordinate ( $\phi_S, \theta_S, \psi_S$ ) could be used. Similar set of equations holds for the car too.

Using the substitution:

$$T_1 = \frac{1}{T_L} + \frac{1}{T_a} + \frac{1}{T_4},\ T_2 = \frac{1}{T_L T_4} + \frac{1}{T_L T_a} + \frac{1}{T_a T_4},\ T_3 = \frac{1}{T_L T_a T_4} \qquad (9.37)$$

the transfer function of the semicircular canals where the sensed rotation is expressed as a function of the angular displacement can be written as:

$$\frac{\hat{\omega}}{\beta} = \frac{\frac{1}{T_4} s^3}{s^3 + T_1 s^2 + T_2 s + T_3} \qquad (9.38)$$

with representation in time domain:

$$\dddot{\hat{\omega}} + T_1 \ddot{\hat{\omega}} + T_2 \dot{\hat{\omega}} + T_3 \hat{\omega} = \frac{1}{T_4} \ddot{\omega} \qquad (9.39)$$

Because the driver frame is parallel to the simulator frame and the pilot's head is located at the origin of the simulator frame, the acceleration at frame *S* is equal to the acceleration at frame *F*.

$$\underline{a}_{PS} = \underline{a}_{SS} = \underline{L}_{IS}\, \underline{a}_{SI} \qquad (9.40)$$

The transformation matrix is given in Equation ( 3.7 ) and by linearizing about $\underline{a}_{SI} = \underline{0}$ and $\underline{\beta}_S = \underline{0}$ one can write:

$$\underline{L}_{IS} = \begin{bmatrix} 1 & \psi_S & -\theta_S \\ -\psi_S & 1 & \phi_S \\ \theta_S & -\phi_S & 1 \end{bmatrix} \qquad (9.41)$$

$$\underline{a}_{PS} = \begin{bmatrix} \underline{a}^x_{SI} + \psi_S \underline{a}^y_{SI} - \theta_S \underline{a}^z_{SI} \\ -\psi_S \underline{a}^x_{SI} + \underline{a}^y_{SI} + \phi_S \underline{a}^z_{SI} \\ \theta_S \underline{a}^x_{SI} - \phi_S \underline{a}^y_{SI} + \underline{a}^z_{SI} \end{bmatrix} \approx \begin{bmatrix} \underline{a}^x_{SI} \\ \underline{a}^y_{SI} \\ \underline{a}^z_{SI} \end{bmatrix} = \underline{a}_{SI} \qquad (9.42)$$

In Equation ( 6.16 ) the computation of the specific force was given, which can be noted again:

$$\underline{f}_{PS} = \underline{a}_{PS} - \underline{L}_{IS} \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = \underline{a}_{PS} - g \begin{bmatrix} -\theta_S \\ \phi_S \\ 1 \end{bmatrix} \qquad (9.43)$$

From here the sensed specific force can be represented taking the otoliths model in Equation ( 4.9 ):

$$\frac{\hat{f}}{f} = \frac{K(\tau_a s + 1)}{(\tau_L s + 1)(\tau_S s + 1)} \qquad (9.44)$$

where *f* stands for a component of the specific force in the simulator or in the car.

Taking the state-space model for the semicircular canal given in Equations ( 4.4 ) - ( 4.6 ) and the substitutions given in Eqiation ( 9.37 ) and relating it to the inputs given in Equation ( 9.31 ) we can write:

$$\dot{\underline{x}} = \underline{A}_{SCC}\,\underline{x} + \underline{B}_{SCC}\,\underline{u} \qquad (9.45)$$

$$\underline{y} = \underline{C}_{SCC}\,\underline{x} + \underline{D}_{SCC}\,\underline{u} \qquad (9.46)$$

where

$$\underline{A}_{SCC} = \begin{bmatrix} -T_1^p & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -T_2^p & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -T_3^p & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -T_1^q & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -T_2^q & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -T_3^q & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -T_1^r & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -T_2^r & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -T_3^r & 0 & 0 \end{bmatrix} \qquad (9.47)$$

$$\underline{B}_{SCC} = \begin{bmatrix} -\frac{T_1^p}{T_4^p} & 0 & 0 & 0 & 0 & 0 \\ -\frac{T_2^p}{T_4^p} & 0 & 0 & 0 & 0 & 0 \\ -\frac{T_3^p}{T_4^p} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{T_1^q}{T_4^q} & 0 & 0 & 0 & 0 \\ 0 & -\frac{T_2^q}{T_4^q} & 0 & 0 & 0 & 0 \\ 0 & -\frac{T_3^q}{T_4^q} & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{T_1^r}{T_4^r} & 0 & 0 & 0 \\ 0 & 0 & -\frac{T_2^r}{T_4^r} & 0 & 0 & 0 \\ 0 & 0 & -\frac{T_3^r}{T_4^r} & 0 & 0 & 0 \end{bmatrix} \qquad (9.48)$$

$$\underline{C}_{SCC} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \qquad (9.49)$$

$$\underline{D}_{SCC} = \begin{bmatrix} \frac{1}{T_4^p} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{T_4^q} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{T_4^r} & 0 & 0 & 0 \end{bmatrix} \quad (9.50)$$

The same procedure can be applied on the otolith model given in Equation ( 4.9 ) and confirmed in Equation ( 9.44 ). One need to take into account the equilibrium values that can be found from Equation ( 9.43 ):

$$\underline{f}_{Pa}^{e} = \underline{f}_{PS}^{e} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} \quad (9.51)$$

and then propagated through the otolith model:

$$\hat{\underline{f}}_{Pa}^{e} = \hat{\underline{f}}_{PS}^{e} = \begin{bmatrix} 0 & 0 & -g\frac{K^z \tau_a^z}{\tau_L^z \tau_S^z} \end{bmatrix} \quad (9.52)$$

The system can be now represented with a state-space model:

$$\dot{\underline{x}} = \underline{A}_{OTO}\,\underline{x} + \underline{B}_{OTO}\,\underline{u} \quad (9.53)$$

$$\underline{y} = \underline{C}_{OTO}\,\underline{x} + \underline{D}_{OTO}\,\underline{u} \quad (9.54)$$

One should be careful when defining the matrices in this model, because the specific force depends not only on the translational acceleration input component, but also from the angular displacement. These dependencies were shown in section 6.4 where the Tilt Coordination method was described. Taking Equation ( 6.27 ), a matrix $\underline{V}$ is defined that will correlate to the input values given in Equation ( 9.31 ).

$$\underline{a}_{SS} = \underline{TC}^{-1}\underline{\beta}_S = \begin{bmatrix} 0 & g & 0 \\ -g & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\underline{\beta}_S \quad (9.55)$$

$$\underline{V} = \begin{bmatrix} 0 & g & 0 & 1 & 0 & 0 \\ -g & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (9.56)$$

The other matrices in the model are:

$$\underline{A}_{OTO} = \begin{bmatrix} -\tau_1^x & 1 & 0 & 0 & 0 & 0 \\ -\tau_2^x & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\tau_1^y & 1 & 0 & 0 \\ 0 & 0 & -\tau_2^y & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\tau_1^z & 1 \\ 0 & 0 & 0 & 0 & -\tau_2^z & 0 \end{bmatrix} \quad (9.57)$$

$$\underline{B}_{OTO} = \begin{bmatrix} K^x\tau_a^x\tau_2^x & 0 & 0 \\ K^x\tau_2^x & 0 & 0 \\ 0 & K^y\tau_a^y\tau_2^y & 0 \\ 0 & K^y\tau_2^y & 0 \\ 0 & 0 & K^z\tau_a^z\tau_2^z \\ 0 & 0 & K^z\tau_2^z \end{bmatrix} \underline{V} \qquad (9.58)$$

$$\underline{C}_{OTO} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \qquad (9.59)$$

where:

$$\tau_1 = \frac{1}{\tau_L} + \frac{1}{\tau_S},\ \tau_2 = \frac{1}{\tau_L\tau_S} \qquad (9.60)$$

The total sensed specific force now can be represented by adding the equilibrium values on the output produced from the model given in Equation ( 9.54 ):

$$\underline{\hat{f}}_{Pa} = \underline{y}_a + \underline{\hat{f}}^e_{Pa} \qquad (9.61)$$

$$\underline{\hat{f}}_{PS} = \underline{y}_S + \underline{\hat{f}}^e_{PS} \qquad (9.62)$$

The state space representation for the whole vestibular system was given in Equation ( 9.11 ) and Equation ( 9.12 ) with now defined matrices:

$$\underline{A}_v = \begin{bmatrix} \underline{A}_{SCC} & \underline{0} \\ \underline{0} & \underline{A}_{OTO} \end{bmatrix},\ \underline{B}_v = \begin{bmatrix} \underline{B}_{SCC} \\ \underline{B}_{OTO} \end{bmatrix} \qquad (9.63)$$

$$\underline{C}_v = \begin{bmatrix} \underline{C}_{SCC} & \underline{0} \\ \underline{0} & \underline{C}_{OTO} \end{bmatrix},\ \underline{D}_v = \begin{bmatrix} \underline{D}_{SCC} \\ \underline{D}_{OTO} \end{bmatrix} \qquad (9.64)$$

The system equations from Section 9.1 can be collected leaving only two undetermined matrices, $\underline{A}_d$ and $\underline{B}_d$. The form of these matrices is found from the internal additional variables $\underline{x}_d$.

The state variables $\underline{x}$ are partitioned as:

- $\{x_1 \cdots x_9\}$ - rotational state variables
- $\{x_{10} \cdots x_{15}\}$ - translational state variables
- $\{x_{16} \cdots x_{30}\}$ - internal additional state variables.

The additional variables in the model are states such as linear velocity and displacement of the simulator, plus some additional terms:

$$\begin{aligned} &\left\{\iint \phi_S dt^2, \int \phi_S dt, \iint \theta_S dt^2, \int \theta_S dt, \iint \psi_S dt^2, \int \psi_S dt, \iiint a_{SI}^x dt^3,\right. \\ &\left.\iint a_{SI}^x dt^2, \int a_{SI}^x dt, \iiint a_{SI}^y dt^3, \iint a_{SI}^y dt^2, \int a_{SI}^y dt, \iiint a_{SI}^z dt^3, \iint a_{SI}^z dt^2, \int a_{SI}^z dt\right\} \end{aligned} \qquad (9.65)$$

The derivation of OWF is done in four modes (pitch/surge, roll/sway, heave and yaw) separately (Appendix F.3 - F.6). Only a subset of the state space variables is used in each mode. This is shown in Table F.1.

## *9.3. Optimization of OWF by using EA*

Although OWF is optimized by the Riccati Solver (Appendix D.2), still there is a great number of weights that need to be tuned (Table F.3). The Riccati Solver finds the local optima (stationary points) very quickly, but we are trying to find the global optimum. By tuning the weights on the matrices *Q, R* and *Rd* we are changing the starting point from which the Riccati Solver starts its survey to the local optima. What one can do in these cases is start the algorithm from many different start points which will produce many locally optimal solutions. From all the solutions only the best will be taken. There exist many start points in the search space and investigating each of them is not possible. For this reason a probabilistic approach is used in which the start points are chosen by heuristic algorithm, i.e. Evolutionary Algorutms.

A heuristic algorithm is one which produces a feasible and hopefully very good but not necessarily optimal solution. Heuristics is generally very fast and efficient. The heuristic is used when an exact algorithm is not available or when it is computationally impractical. Iterative algorithms that can not be proved to converge to optimum solution constitute the class of heuristics.

EA is customized in the similar way as it was done in the CWF optimization, with slight difference in the higher mutation probability (Table F.2). Low mutation probability doesn't help much in this case because there is a big chance that the Riccati Solver can find the same local optimum both for the previous and the mutated individual. The responsibility of the EA is to make probabilistic jumps throughout the search space from where the Riccati Solver starts.

There is another note that needs to be taken. When the system of equations was prepared for the Riccati solver, a filtered white noise was taken as input. This kind of input can cover many driver scenarios, but however, we are mostly interested in optimizing the most common manoeuvres, like acceleration and sudden brake, or periodic lane change. For this reason there are two types of cost estimation. The Riccati Solver is run on each individual with its own values for the matrices *Q, R* and *Rd* (Equation ( 9.20 )) by taking filtered white noise as input, and then EA is run for one of the common manoeuvres as input. With this approach we are considering the optimal solutions that the Riccati Algebraic solver produces, but we are giving advantage on those which give better performance on the standard driving manoeuvres. The procedure goes as follows:

- A population of N individuals is created
- Each individual has its own values for the matrices *Q, R* and *Rd*
- Then in each generation these actions are repeated:
- The Riccati Solver is run on each individual and *W(s)* is estimated
- A simulation is run with the values of *W(s)* and one of our common manoeuvres
- The fitness is found by using the same fitness estimation used in CWF and described in Equation ( 7.23 )
- The ordinary EA procedures are prepared (crossover, mutation, elitism…)
- The population is replaced with its offspring and EA goes through new generation

Similarly like in the CWF optimization, there are four separate modes for the EA optimization on OWF. For the pitch/surge and roll/sway mode 10 variables are optimized, 4 variables for yaw mode and 5 for heave (Table F.4).

# 10. Adaptive Washout Filter (AWF)

The AWF uses feedbacks to adjust its parameters according to the current state of the simulator. The idea is to make full use of the simulator at all time. The development of AWF is based on the work of Telban and Cardullo [ 1 ] and Reid and Nahon [ 7 ] and [ 8 ]. As both authors described, the AWF is applied in $(\ )_I$ components in order to avoid residual motion base displacements. The authors used translational acceleration and angular velocity as input signals, but in this work instead of angular velocity, angular displacement is used. The angular displacement was used in the development of the previous two filters, therefore it is also preferable here.

The implementation of the AWF is given in Figure 10.1. As can be seen, all three channels (translational, rotational and coordination) that were present in the development of CWF still exist and the same logic is preserved. The only difference with CWF is the feedback connections that tune parameters in real time.

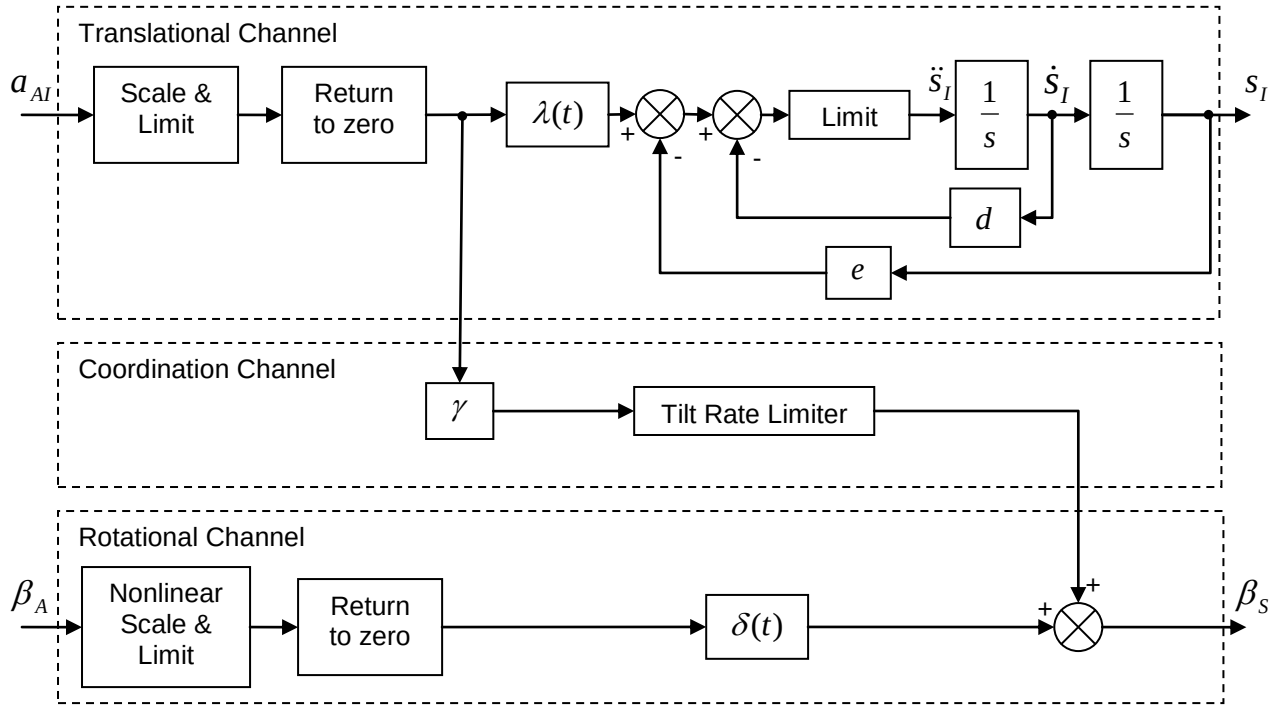


*Figure 10.1: Adaptive washout filter [ 1 ]*

The blocks $\lambda$ and $\delta$ are time varying gains that are continuously adjusted in order to minimize the error given in Equation ( 7.24 ). The gains *e, d* and $\gamma$ have fixed values.

## *10.1. Pitch/Surge Mode*

The pitch angular displacement and the surge translational acceleration can be considered in the following equations:

$$\ddot{s}_I^x = \lambda_x a_{AI}^x - d_x \dot{s}_I^x - e_x s_I^x \qquad (10.1)$$

$$\theta_S = LIM(\gamma_x a_{AI}^x) + \delta_y \theta_A \qquad (10.2)$$

where the first term in Equation ( 10.2 ) comes from tilt coordination.

The gains $\lambda$ and $\delta$ are adjusted so that the cost function is minimized:

$$J_{1x} = \frac{1}{2}\left[\left(a_{AI}^{x} - \ddot{s}_{I}^{x}\right)^2 + w_{x1}\left(\theta_A - \theta_S\right)^2 + w_{x2}\left(\dot{s}_{I}^{x}\right)^2 + w_{x3}\left(s_{I}^{x}\right)^2\right] \qquad (10.3)$$

where $w_{x1}, w_{x2}$ and $w_{x3}$ are constant weights used to penalize the difference between the response in the car and the simulator and to restrain the translational velocity and displacement of the simulator.

The gains $\lambda_x$ and $\delta_y$ are adjusted by steepest descent (Appendix D.3) given as:

$$\dot{\lambda}_x = -G_{\lambda_x}\frac{\partial J_{1x}}{\partial \lambda_x} + K_{\lambda_x}\left(\lambda_{x0} - \lambda_x\right) \qquad (10.4)$$

$$\dot{\delta}_y = -G_{\delta_y}\frac{\partial J_{1x}}{\partial \delta_y} + K_{\delta_y}\left(\delta_{y0} - \delta_y\right) \qquad (10.5)$$

where parameters $G_{\lambda_x}, K_{\lambda_x}, G_{\delta_x}$ and $K_{\delta_x}$ are constants. The left members of the right hand sides of the equations show that the change of the time-varying parameters tends to minimum. The right members cause the parameters $\lambda_x$ and $\delta_y$ return to their neutral position.

## 10.2. Roll/Sway Mode

This mode is very similar to the pitch/surge mode. The description of the parameters is the same, only the coordinates are changed.

$$\ddot{s}_{I}^{y} = \lambda_y a_{AI}^{y} - d_y \dot{s}_{I}^{y} - e_y s_{I}^{y} \qquad (10.6)$$

$$\phi_S = LIM\left(\gamma_y a_{AI}^{y}\right) + \delta_x \phi_A \qquad (10.7)$$

$$J_{1y} = \frac{1}{2}\left[\left(a_{AI}^{y} - \ddot{s}_{I}^{y}\right)^2 + w_{y1}\left(\phi_A - \phi_S\right)^2 + w_{y2}\left(\dot{s}_{I}^{y}\right)^2 + w_{y3}\left(s_{I}^{y}\right)^2\right] \qquad (10.8)$$

$$\dot{\lambda}_y = -G_{\lambda_y}\frac{\partial J_{1y}}{\partial \lambda_y} + K_{\lambda_y}\left(\lambda_{y0} - \lambda_y\right) \qquad (10.9)$$

$$\dot{\delta}_x = -G_{\delta_x}\frac{\partial J_{1y}}{\partial \delta_x} + K_{\delta_x}\left(\delta_{x0} - \delta_x\right) \qquad (10.10)$$

## 10.3. Yaw Mode

For this mode only the yaw angular displacement is considered as input. The cost function is given as:

$$J_2 = \frac{1}{2} w_{z1}\left(\psi_A - \psi_S\right)^2 \qquad (10.11)$$

and the time-varying parameter $\delta_z$ is computed as:

$$\dot{\delta}_z = -G_{\delta_z} \frac{\partial J_2}{\partial \delta_z} + K_{\delta_z}\left(\delta_{z0} - \delta_z\right) \qquad (10.12)$$

## 10.4. Heave Mode

For this mode only the heave acceleration is considered as input. The equations are as follows:

$$\ddot{s}_I^z = \lambda_z a_{AI}^z - d_z \dot{s}_I^z - e_z s_I^z \qquad (10.13)$$

$$J_3 = \frac{1}{2}\left[\left(a_{AI}^z - \ddot{s}_I^z\right)^2 + w_{z2}\left(\dot{s}_I^z\right)^2 + w_{z3}\left(s_I^z\right)^2\right] \qquad (10.14)$$

$$\dot{\lambda}_z = -G_{\lambda_z} \frac{\partial J_3}{\partial \lambda_z} + K_{\lambda_z}\left(\lambda_{z0} - \lambda_z\right) \qquad (10.15)$$

## 10.5. Optimization of AWF by using EA

AWF tunes its parameters $\lambda$ and $\delta$ in real time by using steepest descent optimization. However there are many other parameters (constants) whose values need to be determined (Table G.2). For this reason EA is used with the attributes given in Table G.1 with the common manoeuvres as input. The optimization is run in four separate modes, similarly like in the previous WF optimizations.

The values for $\lambda_0$ and $\delta_0$ are taken to be zero for all of the coordinates. The number of parameters that need to be optimized in each mode is:

- Pitch/surge: 12
- Roll/sway: 12
- Yaw: 4
- Heave: 7

The final values for the parameters are given in Table G.2.

# 11. Nonlinear Washout Filter (NWF)

The Nonlinear WF is described in the work of Telban and Cardullo ([ 1 ], page 97- 130). NWF is a union between AWF and OWF. The idea is to use the Riccati Solver for finding optimal solution as it is done in OFW, but to do this by tuning the filter parameters in real time like it is done in AWF (Figure 11.1). The problem arises because of the slowness of the Riccati Solver. However the authors describe a solution where the Riccati Solver is used to generate result by taking the previous solution as a hint. For this purpose they use structural neural network (Appendix D.1.1).

Only the first initial solution is computed offline and taking this result all successive solutions are found online in real time. NWF is not investigated in this work, but it could be interesting challenge for future researchers.

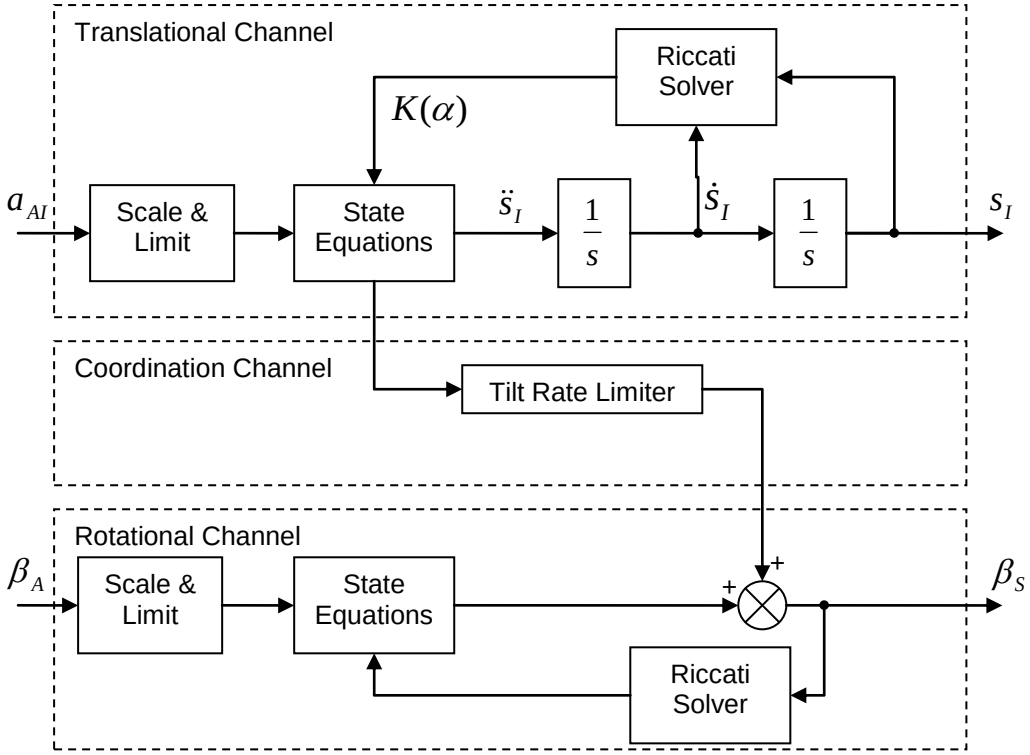


*Figure 11.1: Nonlinear Washout Filter [ 1 ]*

# 12. Results

This work resulted with development of three washout filters, Classical, Optimal and Adaptive. The WFs are modelled in Simulink, Matlab, and their representation is given in Figure E.9, Figure F.7 and Figure G.4 respectively. All WF have similar construction, they all have three channels, translation, coordination and rotation and all three filters expect translational acceleration and angular displacement as input giving the linear and angular displacement as output.

The washout filters are optimized by EA, trying to minimize the difference between the sensation curves from the car and the simulator. For this case a simulation is made that is run for each individual in EA. This Simulink model for the CWF optimization is given in Figure E.11 with its main task to estimate the cost between the sensation curves. Similar models exist for the OWF and AWF optimization. OWF also uses Riccati Solver and AWF uses Steepest Descent method for local optimization.

The same set of input signals is applied on all the filters. The manoeuvres used during optimization are:

- Acceleration from stationary position (ramp to step)
- Acceleration and brake (ramp to step, in positive and negative direction)
- Periodic lane change (used for sway acceleration and yaw displacement)
- Positive and negative acceleration in vertical direction (ramp to step), for heave mode

The WF are also tested with chirp signal and filtered white noise described in Equation ( 9.9 ) with values $\gamma_1 = 0.2$ and $\gamma_2 = 6\pi$. The testes were prepared in computer simulation and the response of the WFs is given in Appendix E.6, F.9 and G.3 for the CWF, OWF and AWF respectively.

The WFs were optimized in four modes, pitch/surge, roll/sway, yaw and heave, separately. The optimal amplitude of the signals that could be simulated in each of these modes was not determined. Instead, the maximum value of the signals that the platform can successfully simulate was found by trial and error. For example, it was found that for pitch/surge mode, the platform can successfully handle surge acceleration of the type given in Figure 12.1 with amplitude up to $2m/s^2$. This is not the optimal amplitude, but that is not of big interest, because when two manoeuvres are combined together, for example surge acceleration and lane change, smaller amplitude than $2m/s^2$ must be used, because otherwise the platform will exceed its physical boundaries, or in other words the inputs must be rescaled to even smaller values.

On the other hand, when yaw mode was tested in real-time on the CVS, the test drivers suggested that the real input was too small so it was rescaled by factor of 2.

After the optimization the final values for the parameters are found given in Table E.4, Table F.6 and Table G.2 for the CWF, OWF and AWF respectively. Taking this configuration for the parameters and surge acceleration as input, the WF gave the response given on Figure 12.1. During the simulation the actuators stayed within its boundaries, but were very close to its limits giving as much of the platform as it can give.

For the sway mode, a periodic lane change manoeuvre was used given on Figure 12.2. The same manoeuvre is used for the yaw mode, but with smaller amplitude, Figure 12.3.

At the end all WFs were tested in computer simulation with a real signal given on Figure C.1 and Figure C.2. The inputs were rescaled proportionally so that the platform stays within its boundaries during the whole simulation. The final scale and limit of the input signals is given in Table E.6, Table F.7 and Table G.3 for the CWF, OWF and AWF respectively.

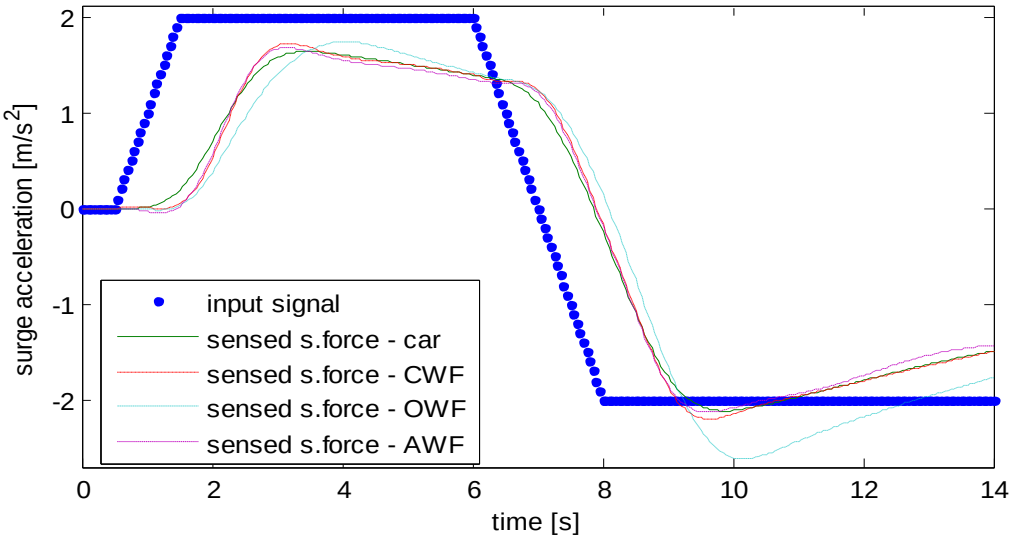


*Figure 12.1: Response of the WFs on surge acceleration*

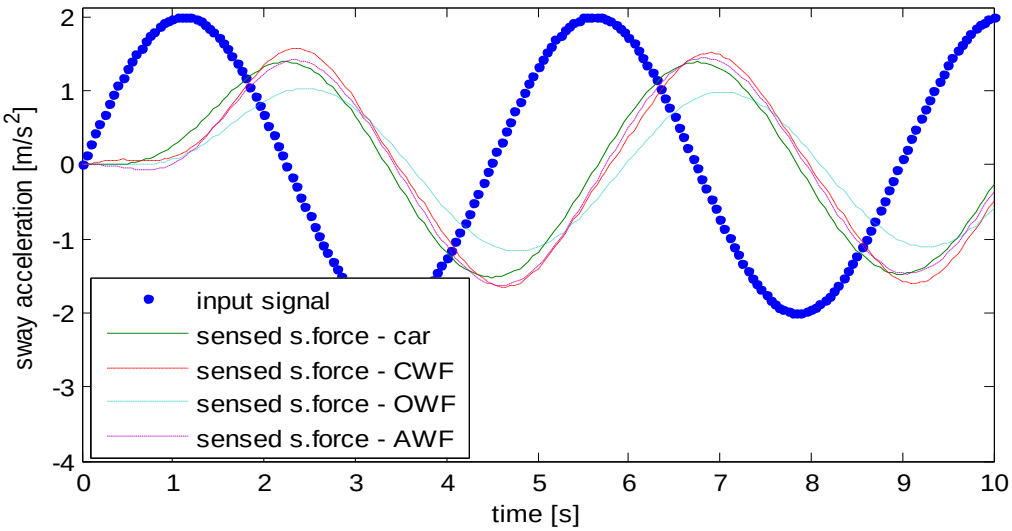


*Figure 12.2: Response of the WFs on sway acceleration*

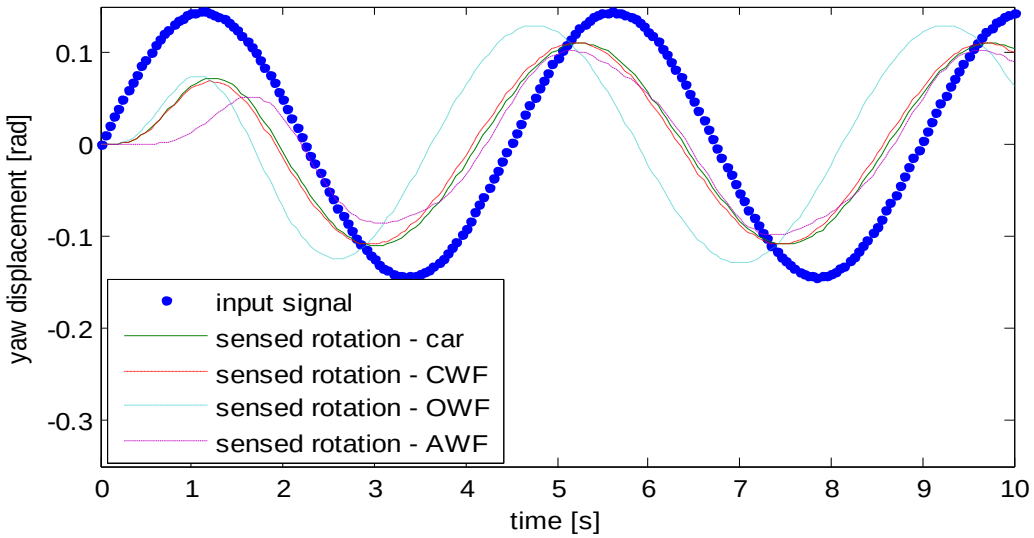


*Figure 12.3: Response of the WFs on yaw displacement*

# 13. Discussion

The optimization of the washout filters was prepared in four separate modes, pitch/surge, roll/sway, yaw and heave. The response of the filters is given in Figure 12.1 - Figure 12.3, and as can be seen all three filters respond very well. Even CWF which has very simple form has very good matching with the reference curve. This is for the reason that CWF can be easily and effectively optimized because of the small number of optimization parameters. AWF gives even better matching than CWF. The response curve is almost not distinguishable from the reference curve. Of course, AWF tunes its parameters in real time and better results were expected.

The response of OWF is not as good as the response of the other two WFs. This is due to the fact that many approximations were used during the estimation of the system of equations. For example, the approximation given in Equation ( 9.35 ) led to conclusion that OWF does not depend on the position of the driver (the driver head more precisely) which contributes to less optimal results. Another problem may be the filtered white noise used as input for the Riccati Solver and not one of the custom manoeuvres. Some of the problems in OWF might emanate from the use of canonical representation since the Riccati Solver is sensitive to the realization. It is preferable that a future thesis worker does the experiments by using balanced realization, e.g. diagonal, block diagonal or Jordan forms. However the final results showed that OWF still gives satisfactory solution.

The matlab function for the Riccati Solver can sometimes generate an error “cannot order eigenvalues; spectrum too near to the imaginary axis”. This error occurs when the Hamiltonian matrix for the Riccati equation has eigenvalues on or very near the imaginary axis. One could resolve this error with small change to any of the weights. Because this error does not happen often and because the process was executed within EA, the simplest way was to skip the individuals that generated the error.

Before the optimization OWF was constructed with four filters: $W_{11}$, $W_{12}$, $W_{21}$ and $W_{22}$. However, after the optimization it was found that the contribution from filter $W_{21}$ is not noticeable and was removed from the final OWF model as can be seen on Figure F.7. OWF was designed for 9$^{th}$ order filters for pitch/surge and roll/sway mode and 5$^{th}$ order for yaw and heave mode. But when the final values for the parameters were found and after cancellations of poles and zeros the final order of the filters was 7$^{th}$ order for pitch/surge and roll/sway mode, 3$^{rd}$ order for yaw and heave mode (coefficients smaller than 0.001 were taken as 0).

CWF and OWF were successfully implemented in real-time on the CVS. But that was not the case with AWF. It happened that the MatlabFcn block does not work in real time and this filter could not be compiled. This block is crucial because it computes derivative of the cost function in respect to the tuning parameters and removing it causes redesign of the whole WF. One could solve this problem by installing newer version of Matlab on the CVS control computers which supports real-time realization of MatlabFcn. The current version used so far, uses Matlab 6.5 and there are several newer versions already. The more complicated solution will require redesign of the AWF in some other way, without including the MatlabFcn block.

The WFs were tested with several test drivers and by their suggestion some changes were made. The inputs were rescaled so that the tilt angle for surge and sway acceleration was lowered and the translational movement was increased. One of the problems here is due to the virtual environment. Large tilt angles cause the virtual environment tilt the car too much in respect to the road, leading in some cases the driver look at the sky or in the road. This was especially noticeable when the driver performs sudden brake at very high speed. The tilt created from sway acceleration was also lowered because the driver was getting the feeling like sailing in a boat. Although really high sway accelerations are recorded, they needed to be lowered, because the driver still felt the tilt from its back, from the contact with the seat.

All the comparisons given in Figure 12.1 - Figure 12.3 are simulated on a computer. It would have been really nice if these comparisons were tested on the actual CVS as well. However at the time this thesis was prepared, the real-time position of the CVS could not have been recorded. These analyses must be left for future researchers after the CVS is upgraded.

Another attempt that could give strength to the importance of the WFs is testing how the platform behaves with or without WF. But the platform can not work at all without WF and these tests can not be done as well (imagine small constant acceleration coming as input on the platform. Without WF, this acceleration, no matter how small it is, will cause the displacement of the platform from its origin to grow exponentially. It is only a matter of time when the platform will exceed its physical limits). One could limit the signals, but the limitation must be done on the platform displacement, and not on the acceleration. Even with limitation it is questionable what should be measured, since the real-time position of the CVS could not be obtained.

The only thing we can rely on is taking conclusions of the computer simulations and most importantly from the test driver's advices.

# 14. Conclusion and Future Work

This work resulted in development of three washout filters from which two were successfully tested in real time use on the Chalmers Vehicle Simulator. Moreover, optimization algorithms were presented for achieving optimal solution and a union of functional and heuristic optimization was given using the power of the both approaches.

CWF and AWF gave very close matching to the reference curves, but OWF produced slightly different results, though still acceptable. When CWF and OWF were tested online, the test drivers did not feel the difference. However one could repeat the derivation of the system of equations for OWF without the approximations used in the current model. This will result in a WF which will better match the reference curve.

The quality of the WFs directly depends on the vestibular model, because they are tuned to respond as close as possible to this model. Improving the vestibular model can produce better results. There already exist models of the vestibular system which are better than the one used in this work [ 1 ]. There are also models which have integrated model of visual sensation, complementing the motion perception [ 1 ]. One should repeat the experiments by using these new models, which will result in more realistic motion.

It was shown that CWF, which was the simplest model of WF, can produce very well results. One should develop another instance of CWF with one order higher filters for achieving even better results.

Washout filters that use real time tuning can give truly promising results, and this was already shown with the Adaptive WF. Therefore, the development of the Nonlinear WF should be expected from the future thesis workers.

A research is recommended for a predictive WF. Using the knowledge about future input signals could minimize the delay in the platform response.

The scale and limit block is present in all three WFs. One could replace this block by nonlinear scaling with input/output characterises briefly given in Appendix E.2. The idea is to use higher gain for small input signals, so that they will be carried above the driver's perception threshold and smaller scaling for high signals to prevent actuator extension over their limits.

# 15. References


*[ 1 ]* Telban, R. J., and Cardullo, F. M., Motion Cueing Algorithm Development: *Human-Centered Linear and Nonlinear Algorithms*. 2005, NASA CR-2005-213747, NASA Langley Research Center, Hampton, VA.

*[ 2 ]* Brian L. Day and Richard C. Fitzpatrick, *The Vestibular System*, Magazine R583

*[ 3 ]* M.Camposaragna , F.Casolo, B.Cattaneo M.Cocetta, *Design and Simulated Test of a Low-Cost Driving Simulator,* ISCSB 2005

*[ 4 ]* Avizzano C.A., Barbagli F., Bergamasco M., *Washout Filter Design for a Motorcycle Simulator*, Proc. Of IEEE SMC2000, Nashville, U.S.A. 8-11 October 2000, pages 995-1000.

*[ 5 ]* Guide to the Motionbase motion cuing algorithms, Motionbase (Holdings) Ltd 1999

*[ 6 ]* Pieter van Balen*, Development of Washout Filters for The Chalmers Vehicle Simulator*, Chalmers University of Technology, Gothenburg, Sweden

*[ 7 ]* L. D. Reid and M. A. Nahon, Flight Simulation Motion-base Drive Algorithms: Part 1 – Developing and Testing the Equations, UTIAS Report No. 296, Univerisy Of Toronto 1985

*[ 8 ]* L. D. Reid and M. A. Nahon, Flight Simulation Motion-base Drive Algorithms: Part 2 – Selecting the System Parameters, UTIAS Report No. 307, Univerisy Of Toronto 1986

*[ 9 ]* L. D. Reid and M. A. Nahon, *Flight Simulation Motion-base Drive Algorithms: Part 3 – Pilot Evaluations*, UTIAS Report No. 319, Univerisy Of Toronto 1985

*[ 10 ]* L. D. Reid and M. A. Nahon, *The Simulation of Truck Motions*, UTIAS Report No. 294, CN ISSN 0082-5255, September 1985

*[ 11 ]* Kwakernaak H., Sivan R, *Linear Optimal Control Systems*, New York, John Wiley & Sons, 1972

*[ 12 ]* Etkin B., *Dynamics of Atmospheric Flight*, New York, John Wiley & Sons, 1972

*[ 13 ]* *Continuous Time Filter Design*, Massachusetts Institute of Technology, Department of Mechanical Engineering, 2.161, Signal Processing, Continuous and Discrete

*[ 14 ]* M. Wahde*, Course of Evolutionary Computation*, Chalmers University of Technology, Department of Physics, Complex Adaptive Systems, 2006

*[ 15 ]* M. I. Zelikin*, Control Theory and Optimization I,* Encyclopaedia of Mathematical Sciences, Volume 86, Springer 2000

*[ 16 ]* William T. Reid*, Riccati Differential Equations,* Mathematics in Science and Engineering, Volume 86, Academic Press New York and London, 1972

# Appendix A. Vestibular System

## *A.1 Parameters of the vestibular system*

| | Roll(x) | Pitch(y) | Yaw(z) |
|---|---|---|---|
| $T_L[s]$ | 6.1 | 5.3 | 10.2 |
| $T_S[s]$ | 0.1 | 0.1 | 0.1 |
| $T_a[s]$ | 30 | 30 | 30 |
| $\delta TH[\mathrm{deg}/s]$ | 3.0 | 3.6 | 2.6 |

*Table A.1: Parameters for the semicircular canal [ 3 ]*

| | Surge(x) | Sway(y) | Heave(z) |
|---|---|---|---|
| $\tau_L[s]$ | 5.33 | 5.33 | 5.33 |
| $\tau_S[s]$ | 0.66 | 0.66 | 0.66 |
| $\tau_a[s]$ | 13.2 | 13.2 | 13.2 |
| $K$ | 0.4 | 0.4 | 0.4 |
| $dTH[m/s^2]$ | 0.17 | 0.17 | 0.28 |

*Table A.2: Parameters for the otoliths [ 3 ]*

## A.2 Behaviour of the vestibular system

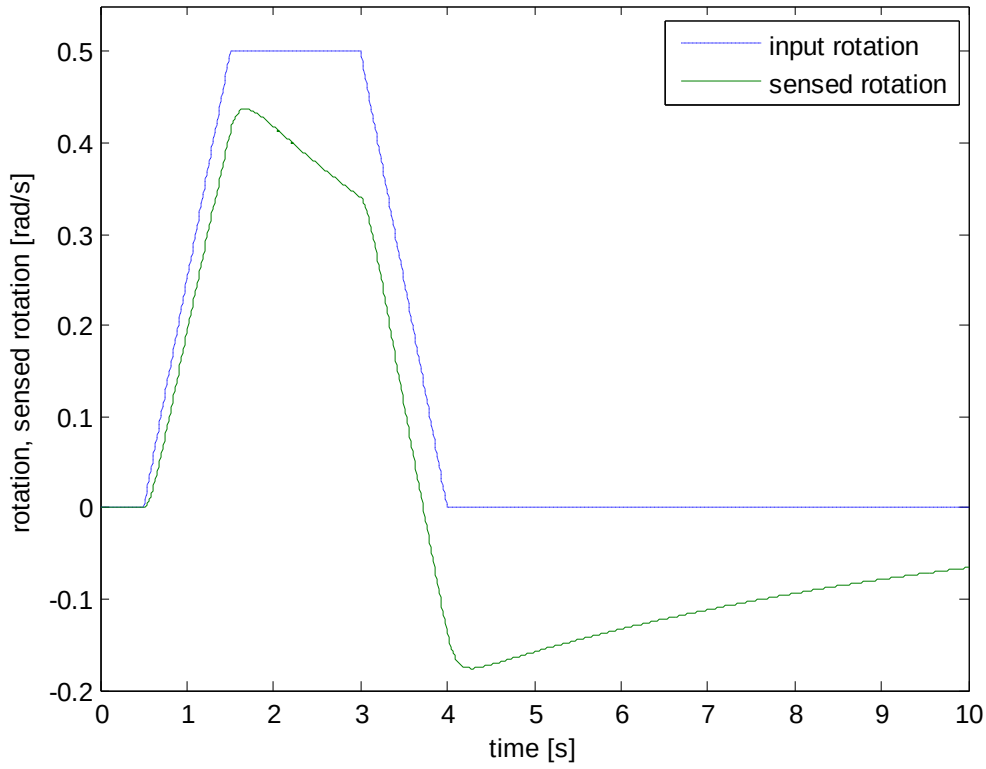


*Figure A.1: Sensed angular velocity for the semicircular canal about the x-axis*

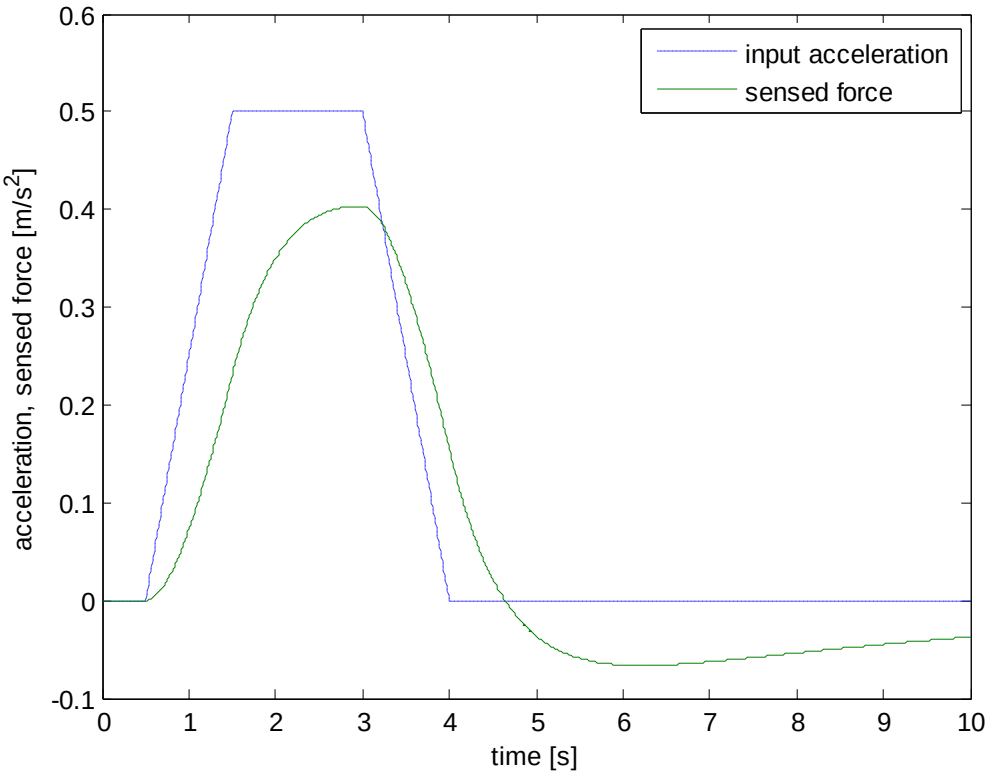


*Figure A.2: Sensed force for the otolith about the x-axis*

# Appendix B. Stewart Platform

## *B.1 Actuator Limits*

| | $X_S$ | $Y_S$ | $Z_S$ |
|---|---|---|---|
| $u_{1S}$ *[m]* | -0.3736 | 0.4442 | 0 |
| $u_{2S}$ *[m]* | -0.1979 | 0.5458 | 0 |
| $u_{3S}$ *[m]* | 0.5715 | 0.1016 | 0 |
| $u_{4S}$ *[m]* | 0.5715 | -0.1016 | 0 |
| $u_{5S}$ *[m]* | -0.1979 | -0.5458 | 0 |
| $u_{6S}$ *[m]* | -0.3736 | -0.4442 | 0 |

*Table B.1: Coordinates of the upper gimbals of the motion platform*

| | $X_I$ | $Y_I$ | $Z_I$ |
|---|---|---|---|
| $b_{1I}$ *[m]* | -0.7874 | 0.1016 | -0.1270 |
| $b_{2I}$ *[m]* | 0.3058 | 07328 | -0.1270 |
| $b_{3I}$ *[m]* | 0.4816 | 0.6312 | -0.1270 |
| $b_{4I}$ *[m]* | 0.4816 | -0.6312 | -0.1270 |
| $b_{5I}$ *[m]* | 0.3058 | -0.7328 | -0.1270 |
| $b_{6I}$ *[m]* | -0.7874 | -0.1016 | -0.1270 |

*Table B.2: Coordinates of the lower gimbals of the motion platform*

| | |
|---|---|
| Mechanical Retract Stop | 0.709*m* |
| Retract Cushion | 0.726*m* |
| Retract Software Limit | 0.739*m* |
| Center Position | 0.879*m* |
| Extend Software Limit | 1.026*m* |
| Extend Cushion | 1.031*m* |
| Mechanical Extend Stop | 1.049*m* |

*Table B.3: Limits of the actuators of the motion platform*

# Appendix C. Motion Cuing

## *C.1 Sample signals from the simulator*

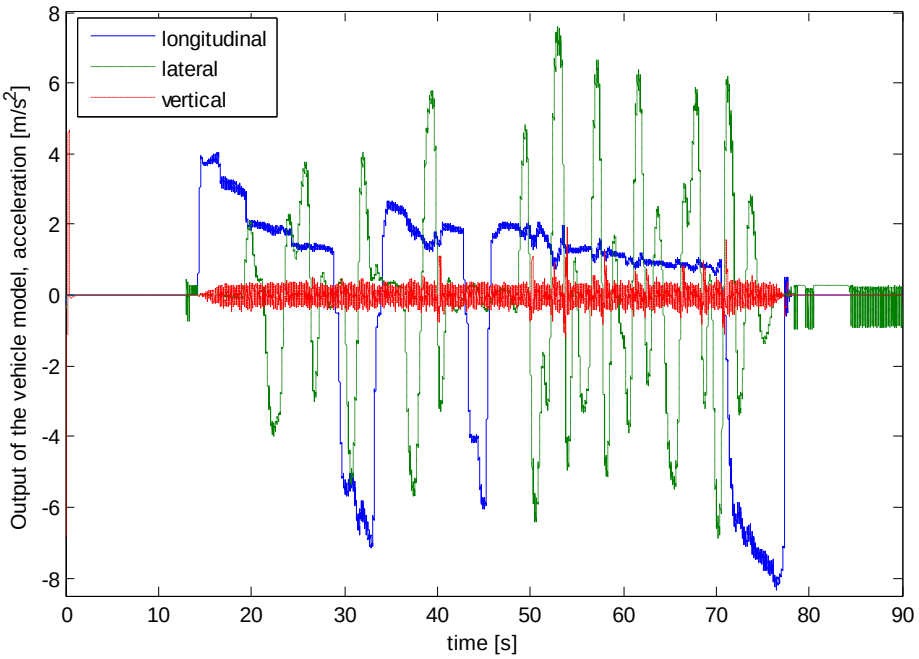


*Figure C.1: Sample acceleration taken from the Matlab software used to control the CVS*

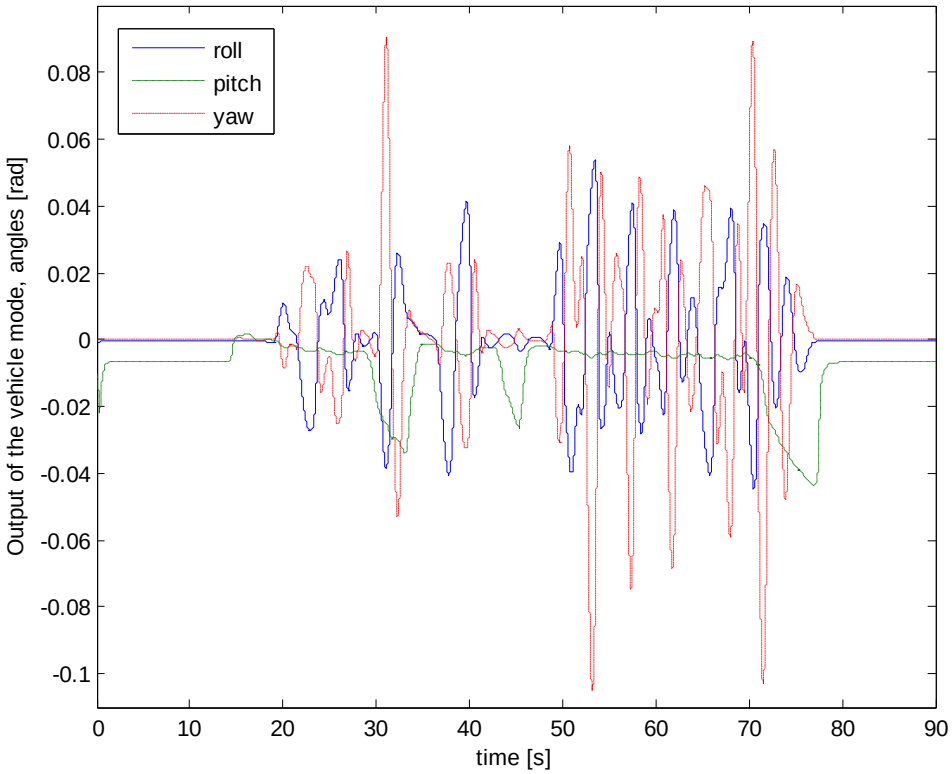


*Figure C.2: Sample angular displacement taken from the Matlab software used to control the CVS*

## C.2 Noise in the signals

Observing the sample data, one could immediately notice the noise in the outputs of the vehicle model, i.e. inputs in the washout filter (Figure C.1). Even in the first five seconds, when the longitudinal acceleration is supposed to have zero value, the noise is still present (Figure C.3). By magnifying the region into the rectangle in Figure C.3 and Figure C.4 one could determine the frequency of the noise-oscillations.

The noise in Figure C.5 is most probably because of the sampling frequency. The sampling time step used in the vehicle model is *0.001s* and the noise given in Figure C.5 has period of *0.002s*, which gives frequency $f = \frac{1}{T} = 500Hz, \omega = 2\pi f$ . However, there exists noise in the signals with lower frequency, that needs to be filtered, i.e. the frequency in Figure C.4 with $\omega_c = 2\pi \frac{1}{0.3} = 6.66\pi$ .

The noise is removed by installing third order Butterworth low-pass filter [ 13 ] with cut-off frequency $\omega_c$ big enough to pass the useful signals and small enough to filter the noise. The value for the cut-off frequency is taken to be $\omega_c = 6\pi$ . The transfer function of the low-pass filter is given in Equation ( C.1 ). One could install the Butterworth filter before the washout filter (WF), but even better case is installing this filter after the WF. This method also filters the eventual HF noise produced by the WF.

$$Gnoise = \frac{\omega_c^3}{s^3 + 2s^2\omega_c + 2s\omega_c^2 + \omega_c^3} \qquad (C.1)$$

The same sample acceleration shown in Figure C.1 is given again in *Figure C.6*, but this time the noise is filtered.

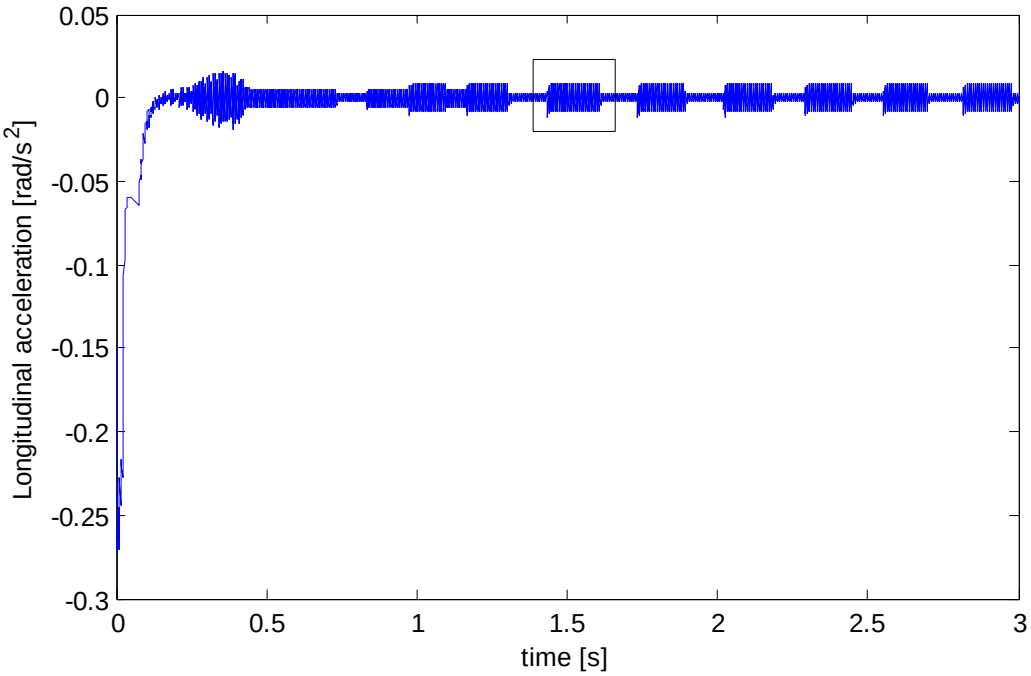


*Figure C.3: Noise in longitudinal acceleration*

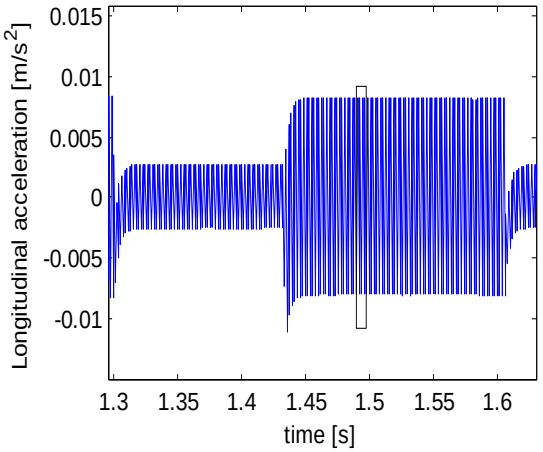


*Figure C.4: Noise (zoom inside the rectangle)*

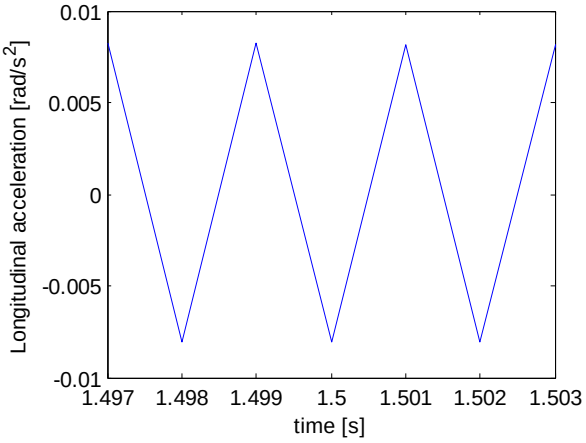


*Figure C.5: Frequency of the noise*

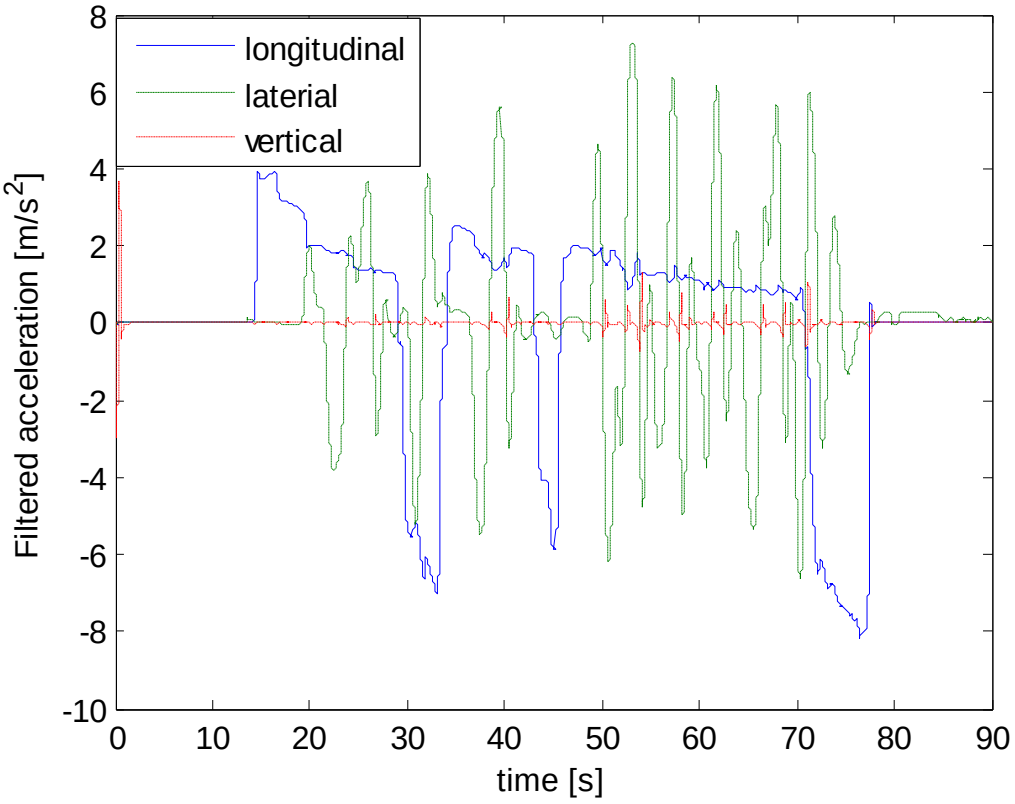


*Figure C.6: Filtered signals*

# Appendix D. Optimization Algorithms

Optimization algorithms are used when optimal solution of some function or process is needed. The necessity of these algorithms can be explained when the domain of possible values in the search space is so large that simple brute force algorithm will require very long time (sometimes more than 1000 years with the standard computer power) to check every possible combination of values. Some of the optimization algorithms have built in logic to narrow straight to the local optimum, like the Newton-Raphson method and the Steepest Descent optimization. Others are using heuristic approach, like the Evolutionary Algorithms. Each algorithm is better than the others for a specific problem, and in this research the Evolutionary Algorithms, Riccati Algebraic Solver and Steepest Descent method are used. A conjunction of these algorithms is also investigated for producing even better results for even shorter time.

## *D.1 Evolutionary Algorithms (EA)*

EAs are motivated from biology and Darwinian evolution is their basic inspiration [ 14 ]. Evolution is a slow process, but this process in nature gave rise to a very complex biological structures. These algorithms are introduced because they have ability to avoid getting stuck in local optima, especially when the search space has many of them, Figure D.1.

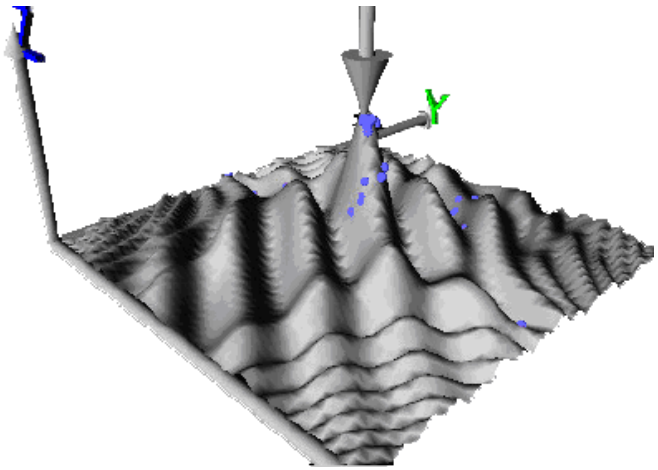

*Figure D.1: Search space with many local optima [ 14 ]*

Genetic Algorithms (GA) are special case of the EA where the values to be optimized are compared to the genes in living beings. In this paper GA are used, but the chromosomes does not have standard array structure and therefore they will be referred only as EA.

Several notions are tightly connected to EA:

- **Population**: a group of the same species, each of them with its own characteristics (genetic mark). Species can mate and can have offspring. The members of the population are referred as **individuals**.
- **Heredity**: the properties of the individual can be encoded in such a way that they can be transmitted to the next **generation**.
- **Fitness**: individuals that are well adapted to its environment have higher fitness.
- **Selection**: the process of selection allows better chance to the best individuals (with higher fitness) survive.
- **Crossover**: the individuals mate by producing offspring that has genetic taken from both of the parents.
- **Mutation**: when genetic information is copied an error can occur. This error is called mutation.

- **Replacement**: the individuals in the population are replaced by new ones in the next generation.
- **Elitism**: even though a fit individual has higher probability of being selected for reproduction, there is no guarantee that it will be selected. Even when it is selected there is a chance it will be destroyed by the crossover. Therefore a few copies of the best individual are copied unchanged into the next generation, and this process is called elitism.

A standard EA goes as follows:

- A population of *N* individuals is generated and default value for the changing parameters is assigned.
- Then in G generations the following process is repeated:
- Each individual is evaluated and the fitness is calculated according to an actual cost function.
- A selection of two individuals is performed.
- These two individuals are mated by crossover, giving offspring of two new individuals.
- The parents are replaced by their offspring.
- Each individual, except the ones under elitism, is mutated under some mutation probability.
- The algorithm continues to the new generation.

## *D.2 Riccati Algebraic Solver*

Riccati differential equations represent one of the simplest types of nonlinear ordinary differential equations and dates from the early period of modern mathematical analyses [ 15 ], [ 16 ]. In mathematics, a scalar Riccati equation is any ordinary differential equation that has the form:

$$y' = q_0(x) + q_1(x)y + q_2(x)y^2 \qquad (D.1)$$

Many different forms and solutions of the Riccati equation exist in theory, but in this work a matrix form is used, given as:

$$\dot{X} = A^T X + XA - XBR^{-1}B^T X + Q = 0 \qquad (D.2)$$

The solution is found by using the matlab function:

$$care(A, B, Q, R) \qquad (D.3)$$

which solves continuous-time algebraic Riccati equation that uses generalized eigen-problem formulation with a Newton-type refinement.

### D.1.1 Real Time Solution of the Riccati Equation

This section is based on the work of Telban and Cardullo [ 1 ] (page 108) and represents a method that solves the Riccati equation in real time. The idea is to use the solution from the previous time step when computing the solution for the current time step. The first solution of the Riccati equation is computed offline, and every other proceeding solution is computed online, in real time.

The authors give two methods: Newton-Raphson approach and a neuro-computing approach using structured neural network. The Newton-Raphson approach uses generalized form of the Riccati equation given in Equation ( D.2 ):

$$G(X) = XSX - XA - A^T X - Q \qquad (D.4)$$

where $S = BR^{-1}B^T$. When the square matrices *X* and *G* are mapped into column vectors:

$$g(X) = \begin{bmatrix} G_{11} & G_{21} & \dots & G_{nn} \end{bmatrix}$$
$$x = \begin{bmatrix} X_{11} & X_{21} & \dots & X_{nn} \end{bmatrix} \qquad (D.5)$$

then it had been shown that the current solution of the Riccati equation can be solved using the previous solution:

$$x(k+1) = x(x) - \left( \frac{\partial G}{\partial X} \left[ x(k) \right] \right)^{-1} g\left[ x(k) \right] \qquad (D.6)$$

where $\frac{\partial G}{\partial X}$ is the Jacobean matrix:

$$\frac{\partial G}{\partial X} = -\left[ I \otimes \left( A - SX \right)^T + \left( A - SX \right)^T \otimes I \right] \qquad (D.7)$$

and $\otimes$ is the Kronecker product.

A structural neural network is used for quickly obtaining the computational solution $X(t)$, Figure D.2.

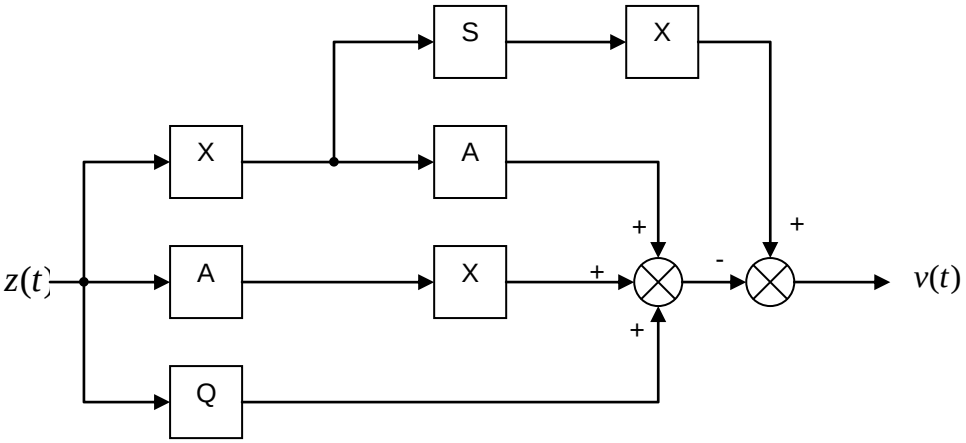


*Figure D.2: Structural Neural Network for solving real time Riccati Equation [ 1 ] (page 110)*

The method of structural Neural Network has some computational advantages over the Newton-Raphson method and it is preferred in future washout filter development. Look at the work of Telban and Cardullo [ 1 ] (page 110) for detailed description.

## D.3 Steepest Descent

The method of steepest descent is also called gradient descent method and it is used to find the nearest local minimum of a function when the gradient of a function is computable.

It starts from point $x_0$ and in each iteration moves from $x_i$ to $x_{i+1}$ by minimizing along the line extending from $x_i$ in the direction of the local downhill gradient $-\nabla f(x_i)$. On Figure D.4 an example is given for one dimensional function:

$$f(x) = x^3 - 2x^2 + 2 \qquad (D.8)$$

where *x* is computed as:

$$x_i = x_{i-1} - \varepsilon f'(x_{i-1}) \qquad (D.9)$$

with $\varepsilon = 0.01$ and starting points $x_0 = 2$ and $x_0 = 0.01$.

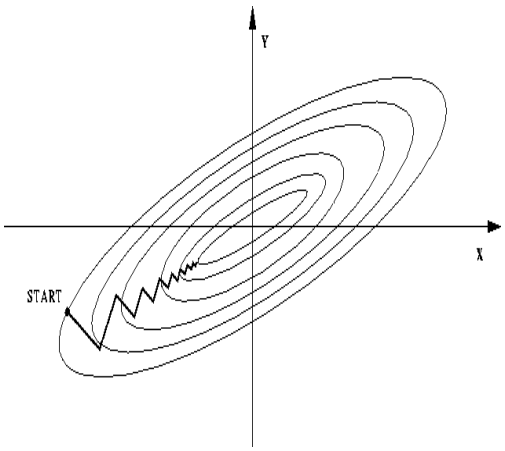


*Figure D.3: Convergence of the Steepest Descent method*

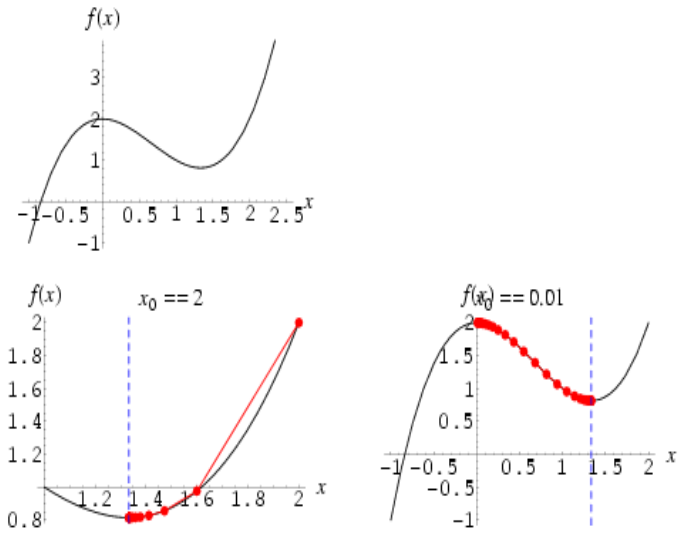


*Figure D.4: Finding minimum with Steepest Descent*
<http://mathworld.wolfram.com/MethodofSteepestDescent.html>

# Appendix E. Classical Washout Filter

## *E.1 Scale and Limit*

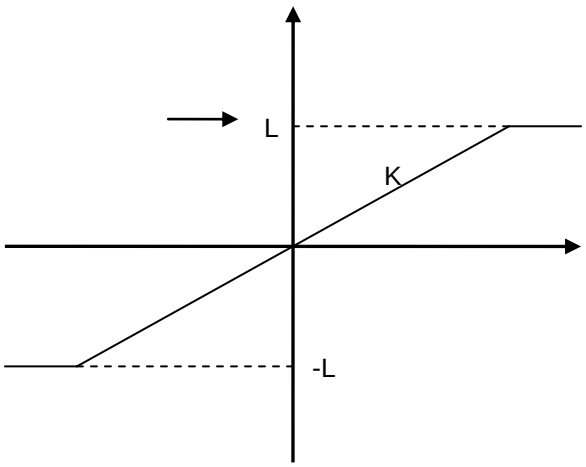


*Figure E.1: Input/output characteristics for scaling and limiting block*

## *E.2 Nonlinear Scale and Limit*

This section is based on the work of Telban and Cardullo ([ 1 ], page 18). A nonlinear scaling is presented as a third-order polynomial:

$$y = c_3 x^3 + c_2 x^2 + c_1 x + c_0 \qquad (E.1)$$

with the input/output characteristics given on Figure E.2.

When the magnitude of the input signal is small, the gain is desired to be higher, or the output may be below the driver's perception threshold. When the output of the input is high, the gain is desired to be smaller so the platform will stay within its boundaries.

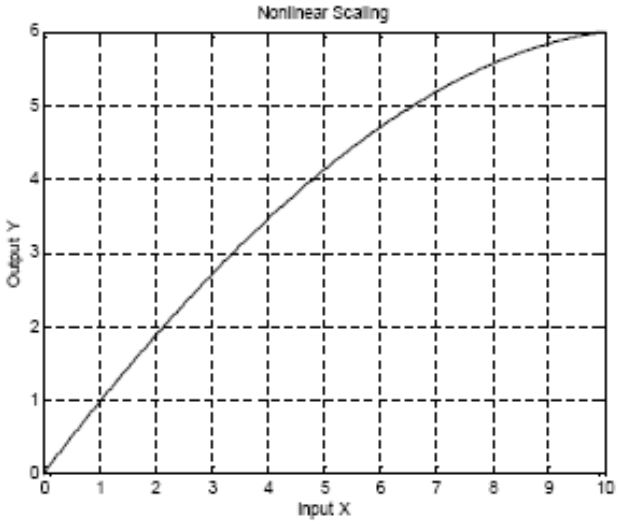


*Figure E.2: Nonlinear input scaling [ 1 ]*

## *E.3 EA attributes for the CWF optimization*

| | |
|---|---|
| Population size | 100 |
| Crossover probability | 0.5 |
| Type of selection | Tournament selection |
| Tournament size | 8 |
| Mutation probability | 0.1 |
| Type of mutation | Geometric creep |
| Creep rate | 0.3 |
| Number of individuals under elitism | 2 |
| Maximum generation | 500 |

*Table E.1: EA attributes for the CWF optimization*

| | | |
|---|---|---|
| OmegaTrans | $\omega_t$ - cut-off frequency for the translational chanel, Equation ( 7.1 ) | Vector of 3 elements, for surge, sway and heave acceleration |
| ZetaTrans | $\zeta_t$ - dumping constant for the translational channel, Equation ( 7.1 ) | Vector of 3 elements, for surge, sway and heave acceleration |
| OmegaTransRTZ | $\omega_b$ - return to zero frequency for the translational channel, Equation ( 7.2 ) | Vector of 3 elements, for surge, sway and heave acceleration |
| OmegaCoord | $\omega_c$ - frequency for the coordination channel, Equation ( 7.4 ) | Vector of 2 elements, for pitch/surge and roll/sway |
| ZetaCoord | $\zeta_c$ - dumping constant for the coordination channel, Equation ( 7.4 ) | Vector of 2 elements, for pitch/surge and roll/sway |
| OmegaRotRTZ | $\omega_b$ - return to zero frequency for the rotational channel, Equation ( 7.5 ) | Vector of 3 elements, for roll, pitch and yaw angle |

*Table E.2: Fields in the chromosome used in EA for the CWF optimization*

| Variable | Pitch/Surge | Roll/Sway | Yaw | Heave |
|---|---|---|---|---|
| OmegaTrans | ✓ | ✓ | | ✓ |
| ZetaTrans | ✓ | ✓ | | ✓ |
| OmegaTransRTZ | ✓ | ✓ | | ✓ |
| OmegaCoord | ✓ | ✓ | | |
| ZetaCoord | ✓ | ✓ | | |
| OmegaRotRTZ | ✓ | ✓ | ✓ | |

*Table E.3: Variables used for separate modes*

## *E.4 Values for the CWF parameters*

| Variable | Values | |
|---|---|---|
| **CWF_W22** | Surge | $\frac{s^3}{s^3 + 20.1065s^2 + 13.1799s + 10.1655}$ |
| | Sway | $\frac{s^3}{s^3 + 11.1047s^2 + 3.6153s + 0.0772}$ |
| | Heave | $\frac{s^3}{s^3 + 11.1047s^2 + 3.6153s + 0.0772}$ |
| **CWF_W12** | Pitch/Surge | $\frac{1222.3}{s^2 + 0.1960s + 1222.3}$ |
| | Roll/Sway | $\frac{1704.4}{s^2 + 7.6094s + 1704.4}$ |
| **CWF_W11** | Roll | $\frac{1}{s + 17.5231}$ |
| | Pitch | $\frac{1}{s + 44.1844}$ |
| | Yaw | $\frac{1}{s + 0.1042}$ |

*Table E.4: Final values for the CWF filters*

| **Translational channel** | **Scale** | **Higher Limit** | **Lower Limit** |
|---|---|---|---|
| Surge $[m/s^2]$ | 1 | 2 | -2 |
| Sway $[m/s^2]$ | 1 | 2 | -2 |
| Heave $[m/s^2]$ | 1 | 2 | -2 |
| **Coordination channel** | **Scale** | **Higher Limit** | **Lower Limit** |
| Pitch/Surge $[m/s^2]$ | 1 | 2 | -2 |
| Roll/Sway $[m/s^2]$ | 1 | 2 | -2 |
| **Rotational channel** | **Scale** | **Higher Limit** | **Lower Limit** |
| Roll *[rad]* | 1 | 0.2 | -0.2 |
| Pitch *[rad]* | 1 | 0.2 | -0.2 |
| Yaw *[rad]* | 1 | 0.2 | -0.2 |

*Table E.5: Values for scale and limit during optimization*

| **Translational channel** | **Scale** | **Higher Limit** | **Lower Limit** |
|---|---|---|---|
| Surge $[m/s^2]$ | 0.45 | 1.8 | -1.8 |
| Sway $[m/s^2]$ | 0.35 | 1.3 | -1.3 |
| Heave $[m/s^2]$ | 0.3 | 0.5 | -0.5 |
| **Coordination channel** | **Scale** | **Higher Limit** | **Lower Limit** |
| Pitch/Surge $[m/s^2]$ | 0.3 | 1.2 | -1.2 |
| Roll/Sway $[m/s^2]$ | 0.2 | 0.8 | -0.5 |
| **Rotational channel** | **Scale** | **Higher Limit** | **Lower Limit** |
| Roll *[rad]* | 1 | 0.2 | -0.2 |
| Pitch *[rad]* | 1 | 0.2 | -0.2 |
| Yaw *[rad]* | 2 | 0.2 | -0.2 |

*Table E.6: Final values for scale and limit*

## *E.5 Bode diagrams of the filters used in CWF*

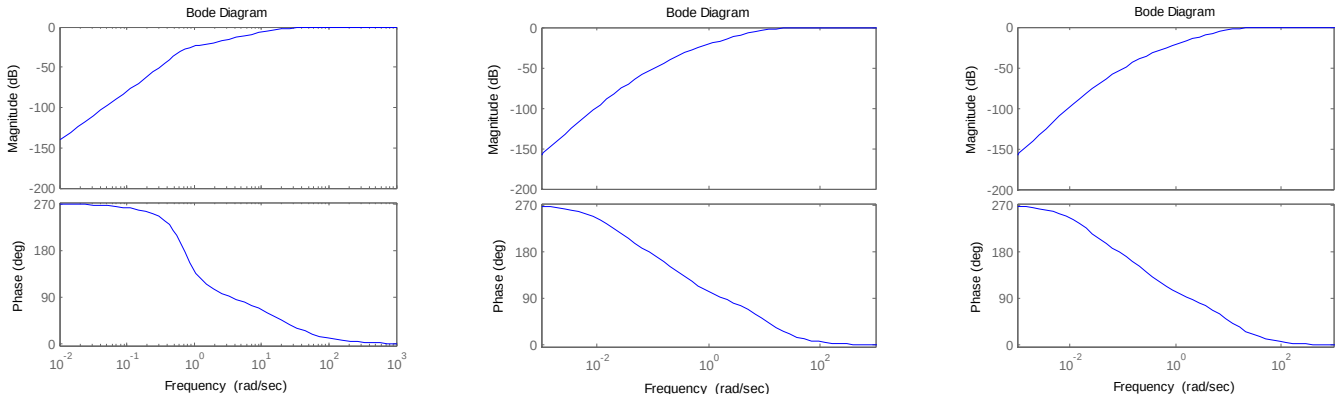


*Figure E.3: Bode plot of the translational filter W22*

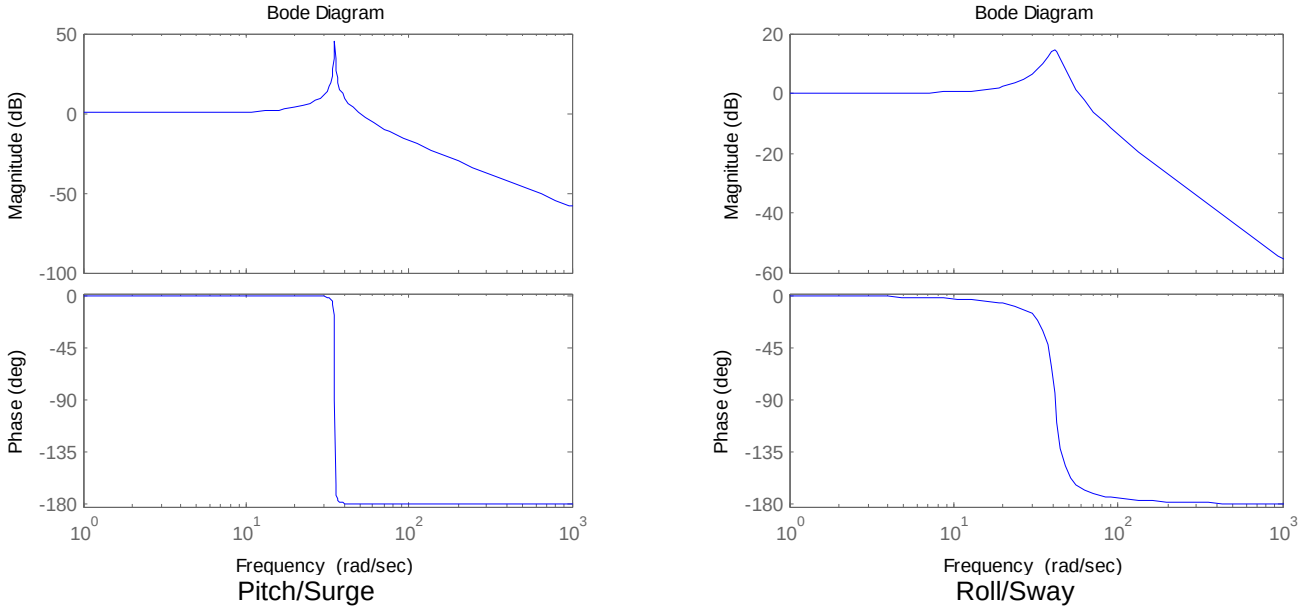


*Figure E.4: Bode plot of the coordination filter W12*

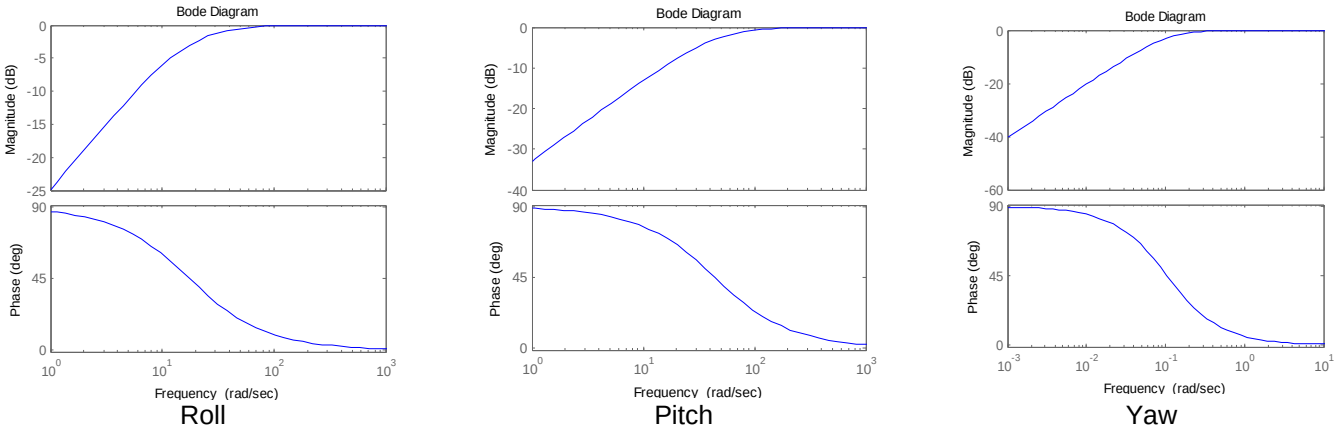


*Figure E.5: Bode plot of the rotational filter W11*

## *E.6 CWF response*

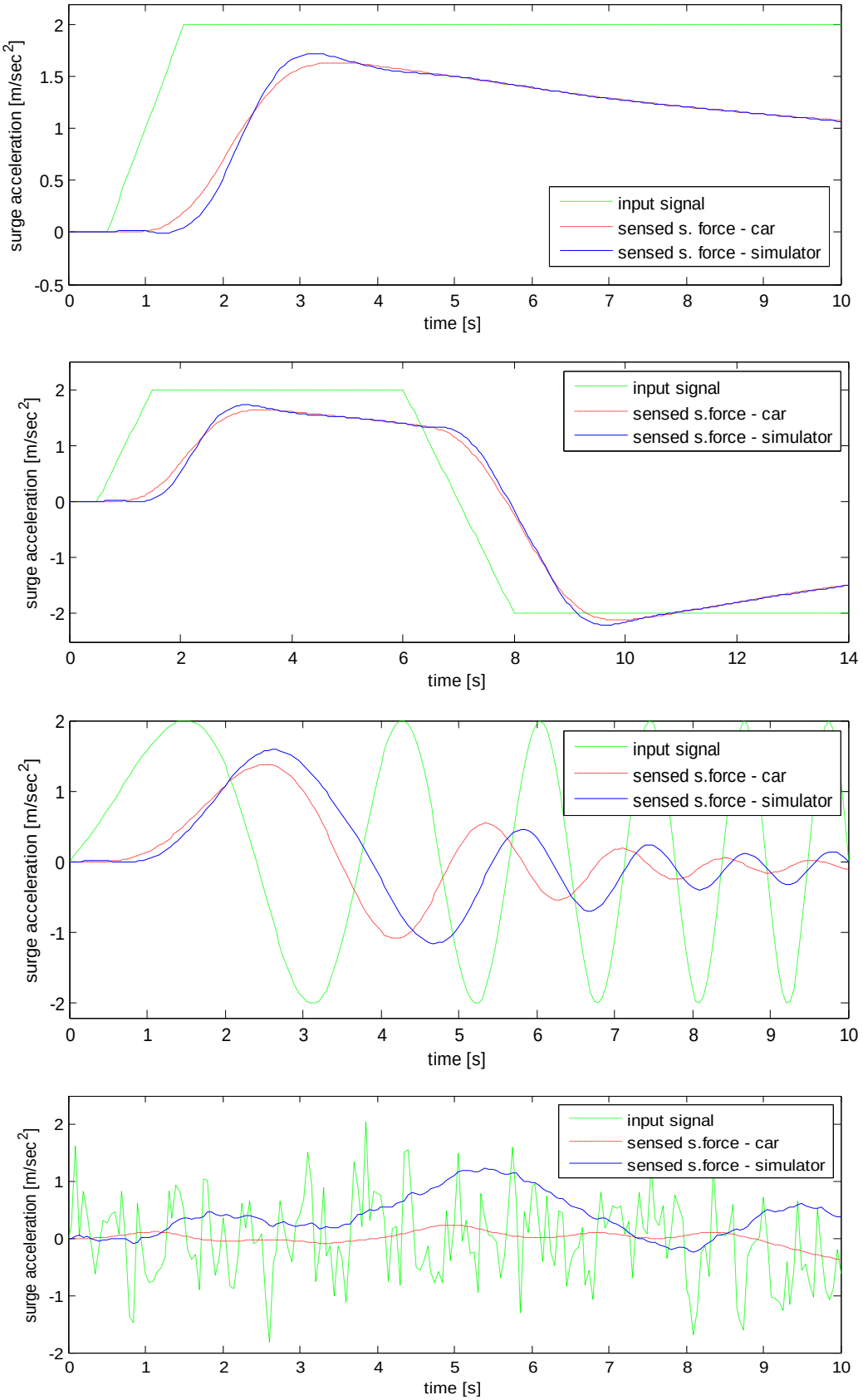


*Figure E.6: CWF response to surge acceleration a) ramp to step of $2m/s^2$, b) sudden acc. and brake, c) chirp, d) filtered white noise (note: s. force = specific force)*

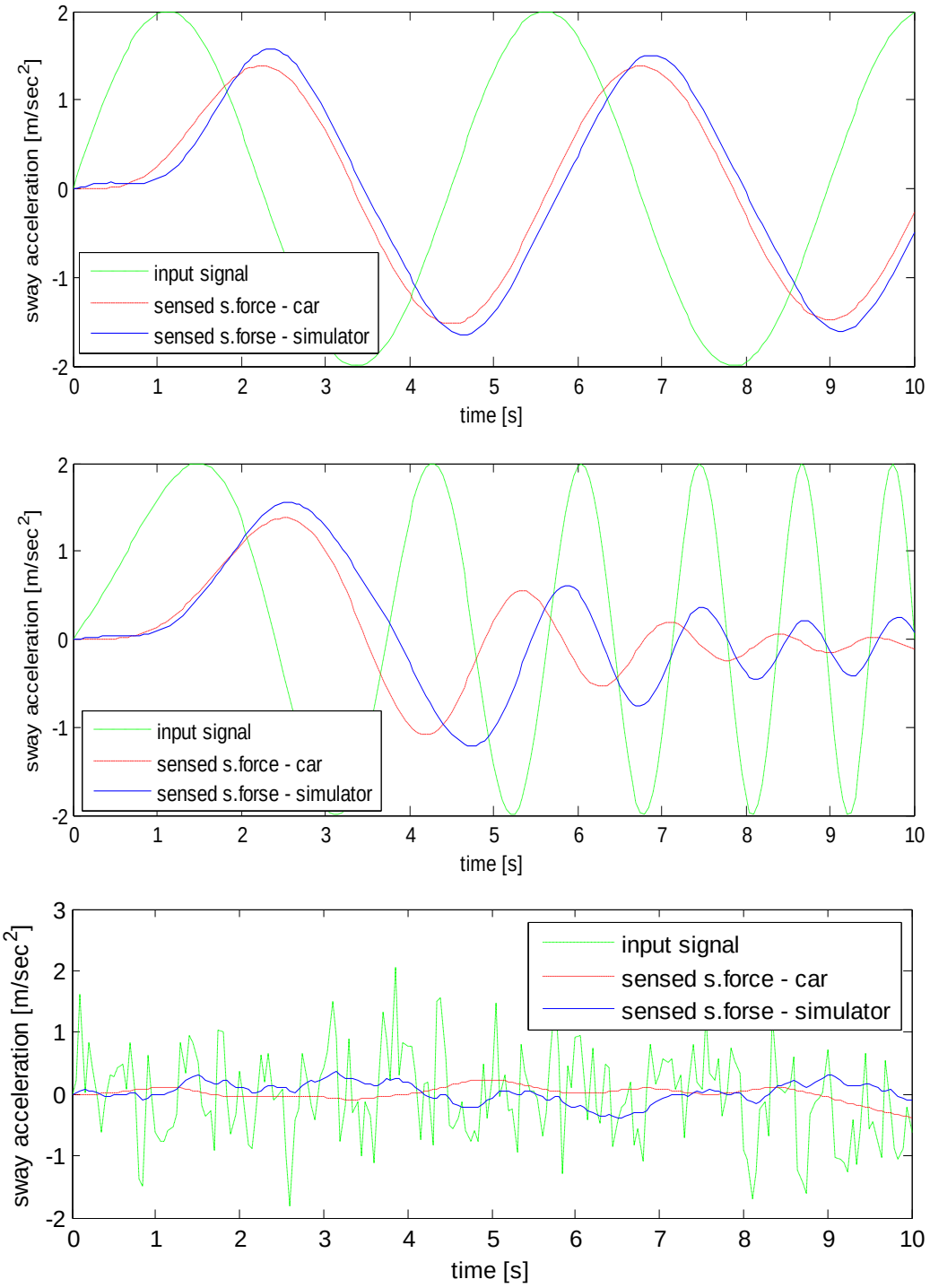


*Figure E.7: CWF response to sway acceleration a ) periodic lane change, b) chirp, c) filtered white noise (note: s. force = specific force)*

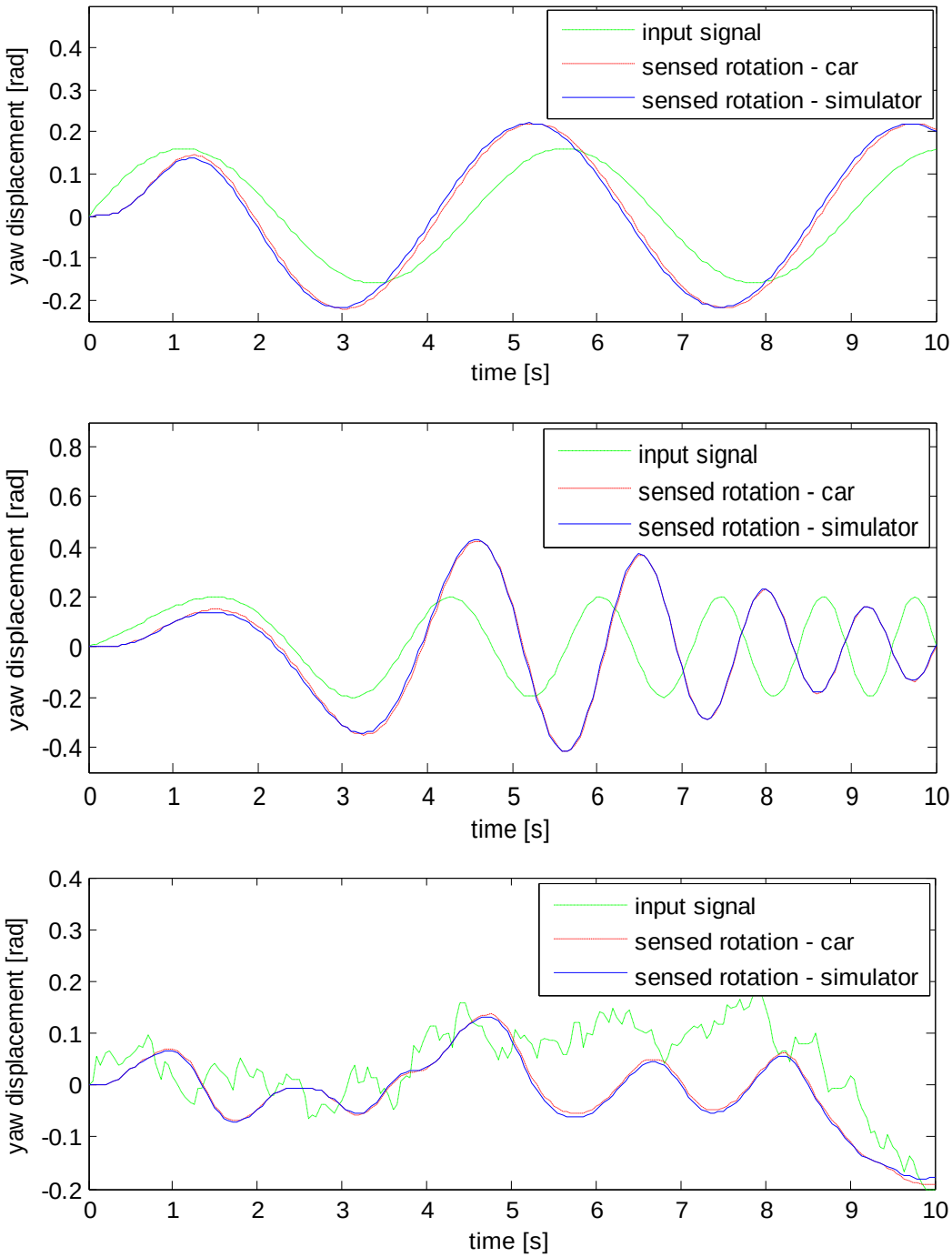


*Figure E.8: CWF response to yaw displacement*
*a) periodic lane change, b) chirp, c) filtered white noise*

## E.7 Simulink models of the CWF

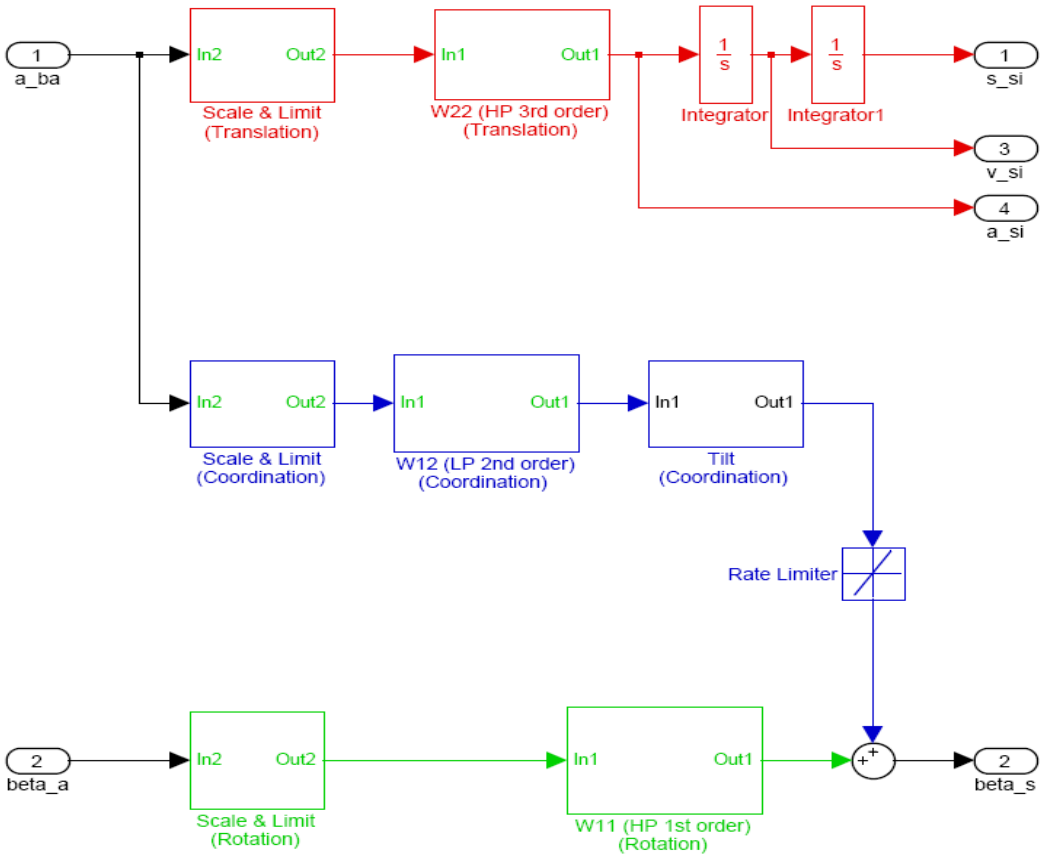


*Figure E.9: Simulink model of the CWF*

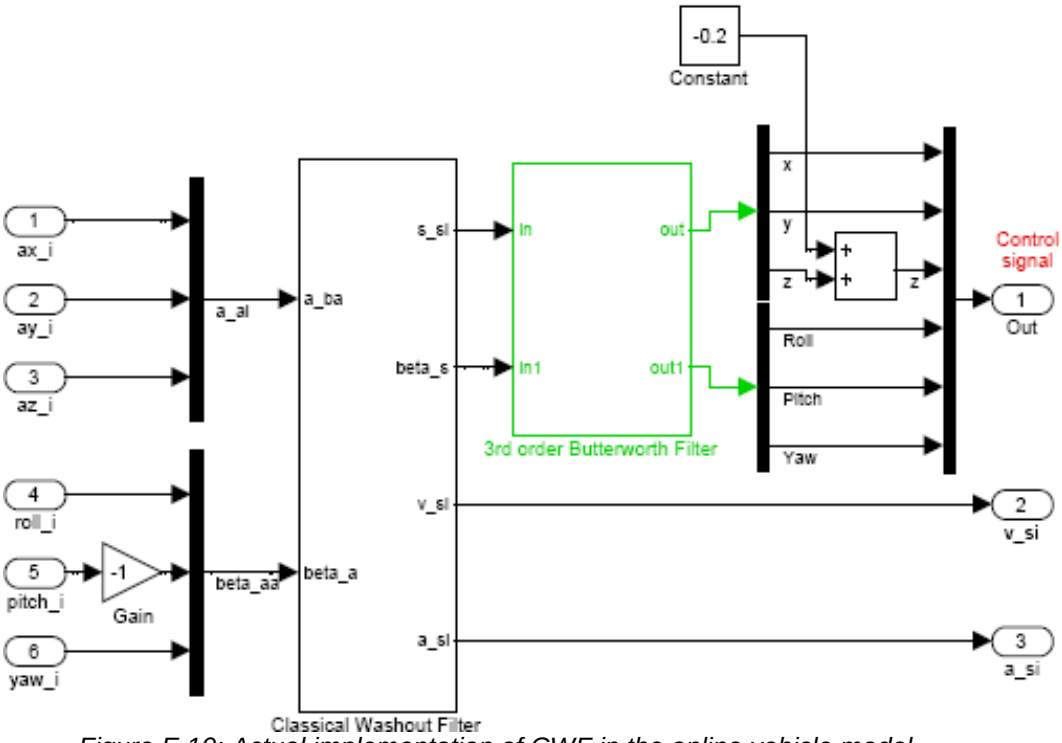


*Figure E.10: Actual implementation of CWF in the online vehicle model (with the Butterworth filter included)*

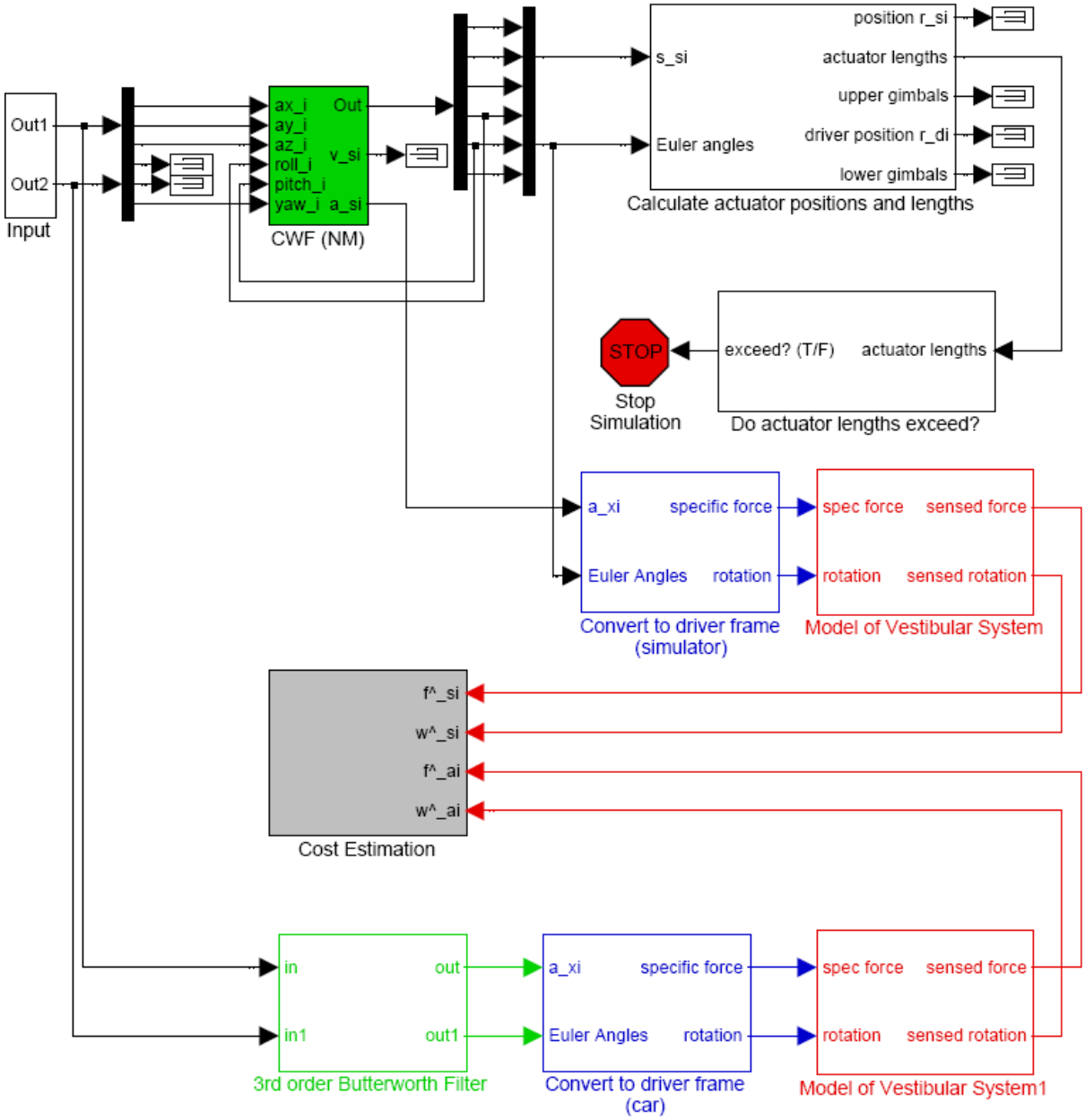


*Figure E.11: Simulink model for cost estimation used by EA*

# Appendix F. Optimal Washout Filter

## *F.1 Cost function for the OWF*

This section is based on the work of Kwakernaak and Sivan [ 11 ] and was noted by Reid and Nahon [ 7 ] (p. H.1 – H.4). A cost function is defined that will constraint both, the sensation error and the platform motion. The goal is minimizing the cost function by selecting suitable control *u*. Consider the system:

$$\dot{\underline{x}} = \underline{A}\underline{x} + \underline{B}\underline{u} + \underline{H}\underline{u}_n \qquad (F.1)$$

$$y = C\underline{x} + \underline{B}\underline{u} \qquad (F.2)$$

where $\underline{u}_n$ is white noise. The cost function is given as:

$$\sigma = E\left\{\int_{t_0}^{t_1}\left[\underline{y}^T\,\underline{G}\underline{y} + \underline{u}^T\,\underline{R}\underline{u}\right]dt\right\} \qquad (F.3)$$

where $\underline{G} \geq 0$, $\underline{R} > 0$ and both matrices are symmetric. The problem can be represented in standard form by several manipulations. Substituting Equation ( 9.3 ) into Equation ( 9.4 ) and using the assumption that the matrices $\underline{G}$ and $\underline{R}$ are symmetric, one could obtain:

$$\sigma = E\left\{\int_{t_0}^{t_1}\left[\underline{x}^T\,\underline{R}_1\,\underline{x} + 2\underline{x}^T\,\underline{R}_{12}\,\underline{u} + \underline{u}^T\,\underline{R}_2\,\underline{u}\right]dt\right\} \qquad (F.4)$$

where:

$$\underline{R}_1 = \underline{C}^T\,\underline{G}\underline{C},\ \underline{R}_{12} = \underline{C}^T\,\underline{G}\underline{D},\ \underline{R}_2 = \underline{R} + \underline{D}^T\,\underline{G}\underline{D} \qquad (F.5)$$

Equation ( F.4 ) can be rearranged in the following form:

$$\sigma = E\left\{\int_{t_0}^{t_1}\left[\underline{x}^T\,\underline{R}'_1\,\underline{x} + \underline{u'}^T\,\underline{R}_2\,\underline{u'}\right]dt\right\} \qquad (F.6)$$

where:

$$\underline{u'} = \underline{u} + \underline{R}_2^{-1}\,\underline{R}_{12}^T\,\underline{x},\ \underline{R}'_1 = \underline{R}_1 - \underline{R}_{12}\,\underline{R}_2^{-1}\,\underline{R}_{12}^T \qquad (F.7)$$

Because $\underline{G}$ and $\underline{R}$ are symmetric, it follows that $\underline{R}_1$ and $\underline{R}_2$ are symmetric too. In similar way it can be shown that $\underline{R}_2 > 0$, $\underline{R}_2^{-1}$ exists and $\underline{R}'_1$ is symmetric.
From Equation ( F.1 ) and ( F.7 ) we have:

$$\dot{\underline{x}} = \underline{A'}\underline{x} + \underline{B}\underline{u'} + \underline{H}\underline{u}_n \qquad (F.8)$$

where:

$$\underline{A'} = \underline{A} - \underline{B}\underline{R}_2^{-1}\,\underline{R}_{12}^T \qquad (F.9)$$

Kwakernaak and Sivan [ 11 ] showed that for this form of problem the optimal control is:

$$\underline{u'} = -\underline{F}(t)\underline{x}\,,\ \underline{F}(t) = \underline{R}_2^{-1}\,\underline{B}^T\,\underline{P}(t) \qquad (F.10)$$

where $\underline{P}(t)$ is solution to the algebraic Riccati Equation:

$$-\dot{\underline{P}} = \underline{R'}_1 - \underline{P}\,\underline{B}\,\underline{R}_2^{-1}\,\underline{B}^T\,\underline{P} + \underline{A'}^T\,\underline{P} + \underline{P}\,\underline{A'}\,,\ P(t_1) = 0 \qquad (F.11)$$

The steady state control law is represented as:

$$\underline{F} = \underline{R}_2^{-1}\,\underline{B}^T\,\underline{P} \qquad (F.12)$$

$$\underline{0} = \underline{R'}_1 - \underline{P}\,\underline{B}\,\underline{R}_2^{-1}\,\underline{B}^T\,\underline{P} + \underline{A'}^T\,\underline{P} + \underline{P}\,\underline{A'} \qquad (F.13)$$

Taking that $\underline{x}(t)$ and $\underline{u}(t)$ are ergodic, Kwakernaak and Sivan [ 11 ] (p. 253-255) showed that the cost function is minimized when:

$$\underline{u} = -\underline{R}_2^{-1}\left[\underline{B}^T\,\underline{P}(t) + \underline{R}_{12}^T\right]\underline{x} \qquad (F.14)$$

$$\sigma = E\left\{\underline{y}^T\,\underline{G}\,\underline{y} + \underline{u}^T\,\underline{R}\underline{u}\right\} \qquad (F.15)$$

## *F.2 State-space variables used in OWF*

| Variable | Description | Pitch/Surge | Roll/Sway | Yaw | Heave |
|---|---|---|---|---|---|
| $x_1$ | rotational state variables (roll) | | ✓ | | |
| $x_2$ | | | ✓ | | |
| $x_3$ | | | ✓ | | |
| $x_4$ | rotational state variables (pitch) | ✓ | | | |
| $x_5$ | | ✓ | | | |
| $x_6$ | | ✓ | | | |
| $x_7$ | rotational state variables (yaw) | | | ✓ | |
| $x_8$ | | | | ✓ | |
| $x_9$ | | | | ✓ | |
| $x_{10}$ | translational state variable (surge) | ✓ | | | |
| $x_{11}$ | | ✓ | | | |
| $x_{12}$ | translational state variable (sway) | | ✓ | | |
| $x_{13}$ | | | ✓ | | |
| $x_{14}$ | translational state variable (heave) | | | | ✓ |
| $x_{15}$ | | | | | ✓ |
| $x_{16}$ | $\iint \phi_S dt^2$ | | | | |
| $x_{17}$ | $\int \phi_S dt$ | | ✓ | | |
| $x_{18}$ | $\iint \theta_S dt^2$ | | | | |
| $x_{19}$ | $\int \theta_S dt$ | ✓ | | | |
| $x_{20}$ | $\iint \psi_S dt^2$ | | | ✓ | |
| $x_{21}$ | $\int \psi_S dt$ | | | ✓ | |
| $x_{22}$ | $\iiint a_{SI}^x dt^3$ | ✓ | | | |
| $x_{23}$ | $\iint a_{SI}^x dt^2$ | ✓ | | | |
| $x_{24}$ | $\int a_{SI}^x dt$ | ✓ | | | |
| $x_{25}$ | $\iiint a_{SI}^y dt^3$ | | ✓ | | |
| $x_{26}$ | $\iint a_{SI}^y dt^2$ | | ✓ | | |
| $x_{27}$ | $\int a_{SI}^y dt$ | | ✓ | | |
| $x_{28}$ | $\iiint a_{SI}^z dt^3$ | | | | ✓ |
| $x_{29}$ | $\iint a_{SI}^z dt^2$ | | | | ✓ |
| $x_{30}$ | $\int a_{SI}^z dt$ | | | | ✓ |

*Table F.1: State space variables used in the development of OWF*

## F.3 Pitch/Surge Mode

A nine order filter is chosen for the pitch/surge mode. The states are given as:

$$\underline{x} = [x_4, x_5, x_6, x_{10}, x_{11}, x_{19}, x_{22}, x_{23}, x_{24}]^T \quad (F.16)$$

where the states $x_4$, $x_5$ and $x_6$ come from the rotational state space variables related to the pitch angle. Variables $x_{10}$ and $x_{11}$ are related to the surge translational state variables and the rest four variables are additional inner state variables given in Table F.1.

The input values in the simulator and the car are:

$$\underline{u}_S = \begin{bmatrix} \theta_S \\ a^x_{SI} \end{bmatrix}, \ \underline{u}_a = \begin{bmatrix} \theta_A \\ a^x_{AI} \end{bmatrix} \quad (F.17)$$

The output values are:

$$\underline{y}_S = \begin{bmatrix} \hat{q}_{PS} \\ \hat{f}^x_{PS} \end{bmatrix}, \ \underline{y}_A = \begin{bmatrix} \hat{q}_{PA} \\ \hat{f}^x_{PA} \end{bmatrix} \quad (F.18)$$

The vestibular system model and the internal variables are given as:

$$\dot{\underline{x}}_v = \underline{A}_v \underline{x} + \underline{B}_v \underline{u}_S \quad (F.19)$$

$$\underline{y}_v = \underline{C}_v \underline{x} + \underline{D}_v \underline{u}_S \quad (F.20)$$

$$\dot{\underline{x}}_d = \underline{A}_d \underline{x}_d + \underline{B}_d \underline{u}_S \quad (F.21)$$

where:

$$\underline{A}_v = \begin{bmatrix} -T_1^q & 1 & 0 & 0 & 0 \\ -T_2^q & 0 & 1 & 0 & 0 \\ -T_3^q & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\tau_1^x & 1 \\ 0 & 0 & 0 & -\tau_2^x & 0 \end{bmatrix}, \ \underline{B}_v = \begin{bmatrix} -\frac{T_1^q}{T_4^q} & 0 \\ -\frac{T_2^q}{T_4^q} & 0 \\ -\frac{T_3^q}{T_4^q} & 0 \\ gK^x\tau_a^x\tau_2^x & K^x\tau_a^x\tau_2^x \\ gK^x\tau_2^x & K^x\tau_2^x \end{bmatrix} \quad (F.22)$$

$$\underline{C}_v = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \ \underline{D}_v = \begin{bmatrix} \frac{1}{T_4^q} & 0 \\ 0 & 0 \end{bmatrix} \quad (F.23)$$

$$x_d = \begin{bmatrix} x_{19} & x_{22} & x_{23} & x_{24} \end{bmatrix}^T = \begin{bmatrix} \int \theta_S dt & \iiint a^x_{SI} dt^3 & \iint a^x_{SI} dt^2 & \int a^x_{SI} dt \end{bmatrix}^T \quad (F.24)$$

The equations for the input noise signal are:

$$\dot{\underline{x}}_n = \underline{A}_n \underline{x}_n + \underline{B}_n \underline{u}_n, \ \underline{u}_a = \underline{x}_n \quad (F.25)$$

Now all the equations for the simulator, the car and the additional inner variables need to be collected into single state space model as was done in Section 9.1. Comparing to Equation ( 9.15 ) - ( 9.17 ) it can be written:

$$\underline{x} = \begin{bmatrix} \underline{x}_e^T & \underline{x}_d^T & \underline{x}_n^T \end{bmatrix}^T \quad (F.26)$$

$$\underline{y} = \begin{bmatrix} \hat{q}_{PS} - \hat{q}_{PA} & \hat{f}_{PS}^x - \hat{f}_{PA}^x & x_{19} & x_{22} & x_{23} & x_{24} \end{bmatrix}^T \quad (F.27)$$

$$\dot{\underline{x}} = \underline{A}\underline{x} + \underline{B}\underline{u}_S + \underline{H}\underline{u}_n \quad (F.28)$$

$$\underline{y} = \underline{C}\underline{x} + \underline{D}\underline{u}_S \quad (F.29)$$

where:

$$\underline{A} = \begin{bmatrix} \underline{A}_v & \underline{0} & -\underline{B}_v \underline{C}_d \\ \underline{0} & \underline{A}_d & \underline{0} \\ \underline{0} & \underline{0} & \underline{A}_n \end{bmatrix},\ \underline{B} = \begin{bmatrix} \underline{B}_v \underline{D}_d \\ \underline{B}_d \\ \underline{0} \end{bmatrix},\ \underline{H} = \begin{bmatrix} \underline{0} \\ \underline{0} \\ \underline{B}_n \end{bmatrix} \quad (F.30)$$

$$\underline{C} = \begin{bmatrix} \underline{C}_v & \underline{0} & -\underline{D}_v \underline{C}_d \\ \underline{0} & \underline{I} & \underline{0} \end{bmatrix},\ \underline{D} = \begin{bmatrix} \underline{D}_v \underline{D}_d \\ \underline{0} \end{bmatrix} \quad (F.31)$$

$$\underline{A}_d = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix},\ \underline{B}_d = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \quad (F.32)$$

The cost function used for minimizing the error was given in Equation ( 9.20 ) and ( 9.21 ):

$$J = E\left\{ \underline{e}^T \underline{Q}\underline{e} + \rho\left( \underline{u}_S^T \underline{R}\underline{u}_S + \underline{x}_d^T \underline{R}_d \underline{x}_d \right) \right\} \quad (F.33)$$

$$\underline{e} = \begin{bmatrix} \hat{q}_{PS} - \hat{q}_{PA} & \hat{f}_{PS}^x - \hat{f}_{PA}^x \end{bmatrix},\ \underline{G} = \begin{bmatrix} \underline{Q} & \underline{0} \\ \underline{0} & \rho\underline{R}_d \end{bmatrix} \quad (F.34)$$

The solution of $\underline{W}(s)$ comes directly from Equation ( 9.30 ).

## F.4 Roll/Sway Mode

Similarly like the pitch/surge mode, a nine order filter is chosen for the roll/sway mode too. The states are given as:

$$\underline{x} = [x_1, x_2, x_3, x_{12}, x_{13}, x_{17}, x_{25}, x_{26}, x_{27}]^T \quad (F.35)$$

where the states $x_1$, $x_2$ and $x_3$ come from the rotational state space variables related to the roll angle. Variable $x_{12}$ and $x_{13}$ are related to the surge translational state variables and the rest four variables are additional inner state variables given in Table F.1.

The input values in the simulator and the car are:

$$\underline{u}_S = \begin{bmatrix} \phi_S \\ a_{SI}^y \end{bmatrix},\ \underline{u}_a = \begin{bmatrix} \phi_A \\ a_{AI}^y \end{bmatrix} \quad (F.36)$$

The output values are:

$$\underline{y}_S = \begin{bmatrix} \hat{p}_{PS} \\ \hat{f}^y_{PS} \end{bmatrix},\ \underline{y}_A = \begin{bmatrix} \hat{p}_{PA} \\ \hat{f}^y_{PA} \end{bmatrix} \qquad (F.37)$$

The vestibular system model and the internal variables are given as:

$$\dot{\underline{x}}_v = \underline{A}_v \underline{x} + \underline{B}_v \underline{u}_S \qquad (F.38)$$

$$\underline{y}_v = \underline{C}_v \underline{x} + \underline{D}_v \underline{u}_S \qquad (F.39)$$

$$\dot{\underline{x}}_d = \underline{A}_d \underline{x}_d + \underline{B}_d \underline{u}_S \qquad (F.40)$$

where:

$$\underline{A}_v = \begin{bmatrix} -T_1^p & 1 & 0 & 0 & 0 \\ -T_2^p & 0 & 1 & 0 & 0 \\ -T_3^p & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\tau_1^y & 1 \\ 0 & 0 & 0 & -\tau_2^y & 0 \end{bmatrix},\ \underline{B}_v = \begin{bmatrix} -\frac{T_1^p}{T_4^p} & 0 \\ -\frac{T_2^p}{T_4^p} & 0 \\ -\frac{T_3^p}{T_4^p} & 0 \\ -gK^y\tau_a^y\tau_2^y & K^y\tau_a^y\tau_2^y \\ -gK^y\tau_2^y & K^y\tau_2^y \end{bmatrix} \qquad (F.41)$$

$$\underline{C}_v = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix},\ \underline{D}_v = \begin{bmatrix} \frac{1}{T_4^p} & 0 \\ 0 & 0 \end{bmatrix} \qquad (F.42)$$

$$x_d = \begin{bmatrix} x_{17} & x_{25} & x_{26} & x_{27} \end{bmatrix}^T = \begin{bmatrix} \int \phi_S dt & \iiint a^y_{SI} dt^3 & \iint a^y_{SI} dt^2 & \int a^y_{SI} dt \end{bmatrix}^T \qquad (F.43)$$

The equations for the input noise signal are:

$$\dot{\underline{x}}_n = \underline{A}_n \underline{x}_n + \underline{B}_n \underline{u}_n,\ \underline{u}_a = \underline{x}_n \qquad (F.44)$$

Now all the equations for the simulator, the car and the additional inner variables need to be collected into single state space model as was done in Section 9.1. Comparing to Equation ( 9.15 ) - ( 9.17 ) it can be written:

$$\underline{x} = \begin{bmatrix} \underline{x}_e^T & \underline{x}_d^T & \underline{x}_n^T \end{bmatrix}^T \qquad (F.45)$$

$$\underline{y} = \begin{bmatrix} \hat{p}_{PS} - \hat{p}_{PA} & \hat{f}^y_{PS} - \hat{f}^y_{PA} & x_{17} & x_{25} & x_{26} & x_{27} \end{bmatrix}^T \qquad (F.46)$$

$$\dot{\underline{x}} = \underline{A}\underline{x} + \underline{B}\underline{u}_S + \underline{H}\underline{u}_n \qquad (F.47)$$

$$\underline{y} = \underline{C}\underline{x} + \underline{D}\underline{u}_S \qquad (F.48)$$

where:

$$\underline{A} = \begin{bmatrix} \underline{A}_v & \underline{0} & -\underline{B}_v \\ \underline{0} & \underline{A}_d & \underline{0} \\ \underline{0} & \underline{0} & \underline{A}_n \end{bmatrix},\ \underline{B} = \begin{bmatrix} \underline{B}_v \\ \underline{B}_d \\ \underline{0} \end{bmatrix},\ \underline{H} = \begin{bmatrix} \underline{0} \\ \underline{0} \\ \underline{B}_n \end{bmatrix} \qquad (F.49)$$

$$\underline{C} = \begin{bmatrix} \underline{C}_v & \underline{0} & -\underline{D}_v \\ \underline{0} & \underline{I} & \underline{0} \end{bmatrix},\ \underline{D} = \begin{bmatrix} \underline{D}_v \\ \underline{0} \end{bmatrix} \qquad (F.50)$$

$$\underline{A}_d = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix},\ \underline{B}_d = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \qquad (F.51)$$

The cost function used for minimizing the error was given in Equation ( 9.20 ) and ( 9.21 ):

$$J = E\left\{\underline{e}^T \underline{Q}\underline{e} + \rho\left(\underline{u}_S^T \underline{R}\underline{u}_S + \underline{x}_d^T \underline{R}_d \underline{x}_d\right)\right\} \qquad (F.52)$$

$$\underline{e} = \begin{bmatrix} \hat{p}_{PS} - \hat{p}_{PA} & \hat{f}_{PS}^{\,y} - \hat{f}_{PA}^{\,y} \end{bmatrix},\ \underline{G} = \begin{bmatrix} \underline{Q} & \underline{0} \\ \underline{0} & \rho\underline{R}_d \end{bmatrix} \qquad (F.53)$$

The solution of $\underline{W}(s)$ comes directly from Equation ( 9.30 ).

## F.5 Yaw Mode

There is only one degree of freedom for the yaw mode, the angular displacement $\psi$. Additional inner states are included concerning movement in *z* direction giving fifth order filter. The states are given as:

$$\underline{x} = [x_7, x_8, x_9, x_{20}, x_{21}]^T \qquad (F.54)$$

where the states $x_7$, $x_8$ and $x_9$ come from the rotational state space variables related to the yaw angle. The rest two variables are additional inner state variables given in Table F.1.

The input values in the simulator and the car are:

$$u_S = \psi_S,\ u_a = \psi_A \qquad (F.55)$$

The output values are:

$$y_S = \hat{r}_{PS},\ y_A = \hat{r}_{PA} \qquad (F.56)$$

The vestibular system model and the internal variables are given as:

$$\dot{\underline{x}}_v = \underline{A}_v \underline{x} + \underline{B}_v u_S \qquad (F.57)$$

$$\underline{y}_v = \underline{C}_v \underline{x} + D_v u_S \qquad (F.58)$$

$$\dot{\underline{x}}_d = \underline{A}_d \underline{x}_d + \underline{B}_d u_S \qquad (F.59)$$

where:

$$\underline{A}_v = \begin{bmatrix} -T_1^r & 1 & 0 \\ -T_2^r & 0 & 1 \\ -T_3^r & 0 & 0 \end{bmatrix},\ \underline{B}_v = \begin{bmatrix} -\frac{T_1^r}{T_4^r} \\ -\frac{T_2^r}{T_4^r} \\ -\frac{T_3^r}{T_4^r} \end{bmatrix}\ \underline{C}_v = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix},\ D_v = \frac{1}{T_4^r} \qquad (F.60)$$

$$x_d = \begin{bmatrix} x_{20} & x_{21} \end{bmatrix}^T = \begin{bmatrix} \iint \psi_S dt^2 & \int \psi_S dt \end{bmatrix}^T \qquad (F.61)$$

The equations for the input noise signal are:

$$\dot{\underline{x}}_n = A_n \underline{x}_n + B_n \underline{u}_n,\ \underline{u}_a = \underline{x}_n \qquad (F.62)$$

Collecting all the equations we have:

$$\underline{x} = \begin{bmatrix} \underline{x}_e^T & \underline{x}_d^T & \underline{x}_n^T \end{bmatrix}^T \qquad (F.63)$$

$$\underline{y} = \begin{bmatrix} \hat{r}_{PS} - \hat{r}_{PA} & x_{20} & x_{21} \end{bmatrix}^T \qquad (F.64)$$

$$\dot{\underline{x}} = \underline{A}\underline{x} + \underline{B}\underline{u}_S + \underline{H}\underline{u}_n \qquad (F.65)$$

$$\underline{y} = \underline{C}\underline{x} + \underline{D}u_S \qquad (F.66)$$

where:

$$\underline{A} = \begin{bmatrix} \underline{A}_v & \underline{0} & -\underline{B}_v \\ \underline{0} & \underline{A}_d & \underline{0} \\ \underline{0} & \underline{0} & \underline{A}_n \end{bmatrix},\ \underline{B} = \begin{bmatrix} \underline{B}_v \\ \underline{B}_d \\ \underline{0} \end{bmatrix},\ \underline{H} = \begin{bmatrix} \underline{0} \\ \underline{0} \\ \underline{B}_n \end{bmatrix} \qquad (F.67)$$

$$\underline{C} = \begin{bmatrix} \underline{C}_v & \underline{0} & -\underline{D}_v \\ \underline{0} & \underline{I} & \underline{0} \end{bmatrix},\ \underline{D} = \begin{bmatrix} D_v \\ \underline{0} \end{bmatrix} \qquad (F.68)$$

$$\underline{A}_d = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix},\ \underline{B}_d = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad (F.69)$$

The cost function used for minimizing the error was given in Equation ( 9.20 ) and ( 9.21 ):

$$J = E\left\{ e^T Q e + \rho\left( u_S^T R u_S + \underline{x}_d^T \underline{R}_d \underline{x}_d \right) \right\} \qquad (F.70)$$

$$e = \hat{r}_{PS} - \hat{r}_{PA},\ \underline{G} = \begin{bmatrix} \underline{Q} & \underline{0} \\ \underline{0} & \rho \underline{R}_d \end{bmatrix} \qquad (F.71)$$

The solution of $\underline{W}(s)$ comes directly from Equation ( 9.30 ).

## F.6 Heave Mode

There is one degree of freedom for the heave mode, the vertical acceleration $a_z$. Additional inner states are included concerning movement in *z* direction giving fifth order filter. The states are given as:

$$\underline{x} = [x_{14}, x_{15}, x_{28}, x_{29}, x_{30}]^T \quad (F.72)$$

where the states $x_{14}$ and $x_{15}$ come from the translational state space variables related to the vertical acceleration. The rest three variables are additional inner state variables given in Table F.1.

The input values in the simulator and the car are:

$$u_S = a_{SI}^z,\ u_a = a_{AI}^z \quad (F.73)$$

The output values are:

$$y_S = \hat{f}_{PS}^z - \hat{f}_{PS}^e,\ y_a = \hat{f}_{PA}^z - \hat{f}_{PA}^e \quad (F.74)$$

where the equilibrium values can be found in Equation ( 9.52 ). The vestibular system model and the internal variables are given as:

$$\dot{x}_v = A_v x + B_v u_S \quad (F.75)$$
$$y_v = C_v x + D_v u_S \quad (F.76)$$
$$\dot{\underline{x}}_d = \underline{A}_d \underline{x}_d + \underline{B}_d u_S \quad (F.77)$$

where:

$$\underline{A}_v = \begin{bmatrix} -\tau_1^z & 1 \\ -\tau_2^z & 0 \end{bmatrix},\ \underline{B}_v = \begin{bmatrix} K^z \tau_a^z \tau_2^z \\ K^z \tau_2^z \end{bmatrix},\ \underline{C}_v = \begin{bmatrix} 1 & 0 \end{bmatrix},\ \underline{D}_v = 0 \quad (F.78)$$

$$x_d = \begin{bmatrix} x_{28} & x_{29} & x_{30} \end{bmatrix}^T = \begin{bmatrix} \iiint a_{SI}^z dt^3 & \iint a_{SI}^z dt^2 & \int a_{SI}^z dt \end{bmatrix}^T \quad (F.79)$$

The equations for the input noise signal are:

$$\dot{\underline{x}}_n = \underline{A}_n \underline{x}_n + \underline{B}_n \underline{u}_n,\ \underline{u}_a = \underline{x}_n \quad (F.80)$$

Collecting all the equations we have:

$$\underline{x} = \begin{bmatrix} \underline{x}_e^T & \underline{x}_d^T & \underline{x}_n^T \end{bmatrix}^T \quad (F.81)$$
$$\underline{y} = \begin{bmatrix} \hat{f}_{PS}^z - \hat{f}_{PA}^z & x_{28} & x_{29} & x_{30} \end{bmatrix}^T \quad (F.82)$$
$$\dot{\underline{x}} = \underline{A}\underline{x} + \underline{B}\underline{u}_S + \underline{H}\underline{u}_n \quad (F.83)$$
$$\underline{y} = \underline{C}\underline{x} + \underline{D}u_S \quad (F.84)$$

where:

$$\underline{A} = \begin{bmatrix} A_v & \underline{0} & -B_v \\ \underline{0} & \underline{A}_d & \underline{0} \\ 0 & \underline{0} & A_n \end{bmatrix},\ \underline{B} = \begin{bmatrix} B_v \\ \underline{B}_d \\ \underline{0} \end{bmatrix},\ \underline{H} = \begin{bmatrix} \underline{0} \\ \underline{0} \\ B_n \end{bmatrix} \quad (F.85)$$

$$\underline{C} = \begin{bmatrix} C_v & \underline{0} & -D_v \\ \underline{0} & \underline{I} & \underline{0} \end{bmatrix},\ \underline{D} = \begin{bmatrix} D_v \\ \underline{0} \end{bmatrix} \quad (F.86)$$

$$\underline{A}_d = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix},\ \underline{B}_d = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad (F.87)$$

The cost function used for minimizing the error was given in Equation ( 9.20 ) and ( 9.21 ):

$$J = E\left\{ e^T Q e + \rho\left( u_S^T R u_S + \underline{x}_d^T \underline{R}_d \underline{x}_d \right) \right\} \qquad (F.88)$$

$$e = \hat{f}_{PS}^{\,z} - \hat{f}_{PA}^{\,z},\ \underline{G} = \begin{bmatrix} Q & \underline{0} \\ \underline{0} & \rho \underline{R}_d \end{bmatrix} \qquad (F.89)$$

The solution of $\underline{W}(s)$ comes directly from Equation ( 9.30 ).

## F.7 EA attributes for the OWF optimization

| | |
|---|---|
| Population size | 50 |
| Crossover probability | 0.7 |
| Type of selection | Tournament selection |
| Tournament size | 8 |
| Mutation probability | 0.3 |
| Type of mutation | Geometric creep |
| Creep rate | 0.7 |
| Number of individuals under elitism | 2 |
| Maximum generation | 500 |

*Table F.2: EA attributes for the OWF optimization*

| | | |
|---|---|---|
| Q | *Q* – weighting matrix on the error | Matrix of 4 rows and 2 columns |
| Rd | Rd – weighting matrix on the additional variables | Matrix of 4 rows and 4 columns |
| R | *R* – weighting matrix on the input | Matrix of 2 rows and 2 columns |

*Table F.3: Fields in the chromosome used in EA for the OWF optimization*

| **Pitch/Surge** | **Roll/Sway** | **Yaw** | **Heave** |
|---|---|---|---|
| diag(Q(1, :)) | diag(Q(2, :)) | Q(3, 1) | Q(4, 2) |
| diag(Rd(1, :)) | diag(Rd(2, :)) | diag([Rd(3, 1) Rd(3, 2)]) | diag([Rd(4, 1) Rd(4, 2) Rd(4, 3)]) |
| diag(R(1, :)) | diag(R(2, :)) | R(3, 1) | R(4, 2) |

*Table F.4: Variables used for separate modes*

| **Variable** | **Pitch/Surge** | **Roll/Sway** | **Yaw** | **Heave** |
|---|---|---|---|---|
| Q(1, 1) | 0.3545 | 8.4942 | 470879.25 | |
| Q(2, 2) | 1.4585 | 32.9538 | | 679.0803 |
| R(1, 1) | 1838.77 | 1556.97 | 900654.76 | |
| R(2, 2) | 15.30 | 13.09 | | 1794.99 |
| Rd(1, 1) | 0.0090 | 7874.78 | 0 | 0.0059 |
| Rd(2, 2) | 0.4082 | 2.2450 | 0.0038 | 0.0108 |
| Rd(3, 3) | 5.5972 | 21.3720 | | 0.1182 |
| Rd(4, 4) | 60.8231 | 3.4843 | | |
| $\rho$ | 1 | 1 | 1 | 1 |
| $\gamma_1$ | 0.2 | 0.2 | 0.2 | |
| $\gamma_2$ | $6\pi$ | $6\pi$ | | $6\pi$ |

*Table F.5: Final parameters for the Riccati Solver*

# *Values for the OWF parameters*

| Variable | Values | |
|---|---|---|
| **OWF_W22** | Surge | $\frac{0.0031s^8+0.1277s^7+0.9984s^6+0.3858s^5+0.0441s^4+0.0011s^3}{s^8+14.3901s^7+51.1219s^6+70.4055s^5+41.7806s^4+12.6622s^3+1.9896s^2+0.1445s+0.0032}$ |
| | Sway | $\frac{0.0053s^7+0.0448s^6+0.0834s^5+0.0273s^4+0.0028s^3}{s^8+15.3536s^7+78.2529s^6+173.4813s^5+197.4819^4+126.5639^3+43.1498s^2+7.3520s+0.5506}$ |
| | Heave | $\frac{0.0103s^4+0.2263s^3+0.0332s^2}{s^4+2.1821s^3+0.7892s^2+0.1285s+0.0119}$ |
| **OWF_W12** | Pitch/Surge | $\frac{0.0079s^8+0.2778s^7+2.4866s^6+5.3406s^5+3.2642s^4+0.8952s^3+0.1115s^2+0.0054s}{s^8+14.3901s^7+51.1219s^6+70.4055s^5+41.7806s^4+12.6622s^3+1.9896s^2+0.1445s+0.0032}$ |
| | Roll/Sway | $\frac{-0.0129s^8-0.5168s^7-4.8423s^6-10.6830s^5-10.5602s^4-4.4572s^3-0.8298s^2-0.0639s-0.0014}{s^8+15.3536s^7+78.2529s^6+173.4813s^5+197.4819^4+126.5639^3+43.1498s^2+7.3520s+0.5506}$ |
| **OWF_W11** | Roll | $\frac{-0.0097s^8-1.1480s^7-1.2740s^6+0.375s^5+2.305s^4+1.2759s^3+0.2461s^2+0.0181s}{s^8+15.3536s^7+78.2529s^6+173.4813s^5+197.4819^4+126.5639^3+43.1498s^2+7.3520s+0.5506}$ |
| | Pitch | $\frac{0.0269s^8+0.3437s^7+1.5885s^6+2.7060s^5+1.6212s^4+0.4606s^3+0.0632s^2+0.0037s}{s^8+14.3901s^7+51.1219s^6+70.4055s^5+41.7806s^4+12.6622s^3+1.9896s^2+0.1445s+0.0032}$ |
| | Yaw | $\frac{0.8779s^3+0.2605s^2+0.0223s}{s^3+1.4979s^2+0.1834s+0.0045}$ |

*Table F.6: Final values for the OWF filters*

| **Translational channel** | **Scale** | **Higher Limit** | **Lower Limit** |
|---|---|---|---|
| Surge $[m/s^2]$ | 0.2 | 0.8 | -0.8 |
| Sway $[m/s^2]$ | 0.1 | 0.4 | -0.4 |
| Heave $[m/s^2]$ | 0.2 | 0.5 | -0.5 |
| **Coordination channel** | **Scale** | **Higher Limit** | **Lower Limit** |
| Pitch/Surge $[m/s^2]$ | 0.3 | 1.2 | -1.2 |
| Roll/Sway $[m/s^2]$ | 0.2 | 0.8 | -0.5 |
| **Rotational channel** | **Scale** | **Higher Limit** | **Lower Limit** |
| Roll *[rad]* | 1 | 0.2 | -0.2 |
| Pitch *[rad]* | 1 | 0.2 | -0.2 |
| Yaw *[rad]* | 1.8 | 0.2 | -0.2 |

*Table F.7: Final values for scale and limit*

## *F.8 Bode diagrams of the filters used in OWF*

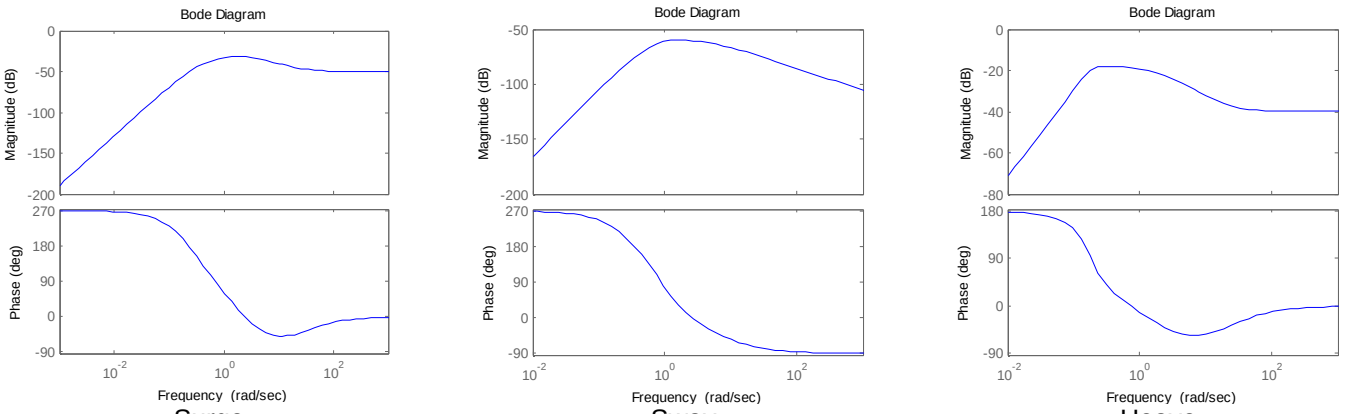


*Figure F.1: Bode plot of the translational filter W22*

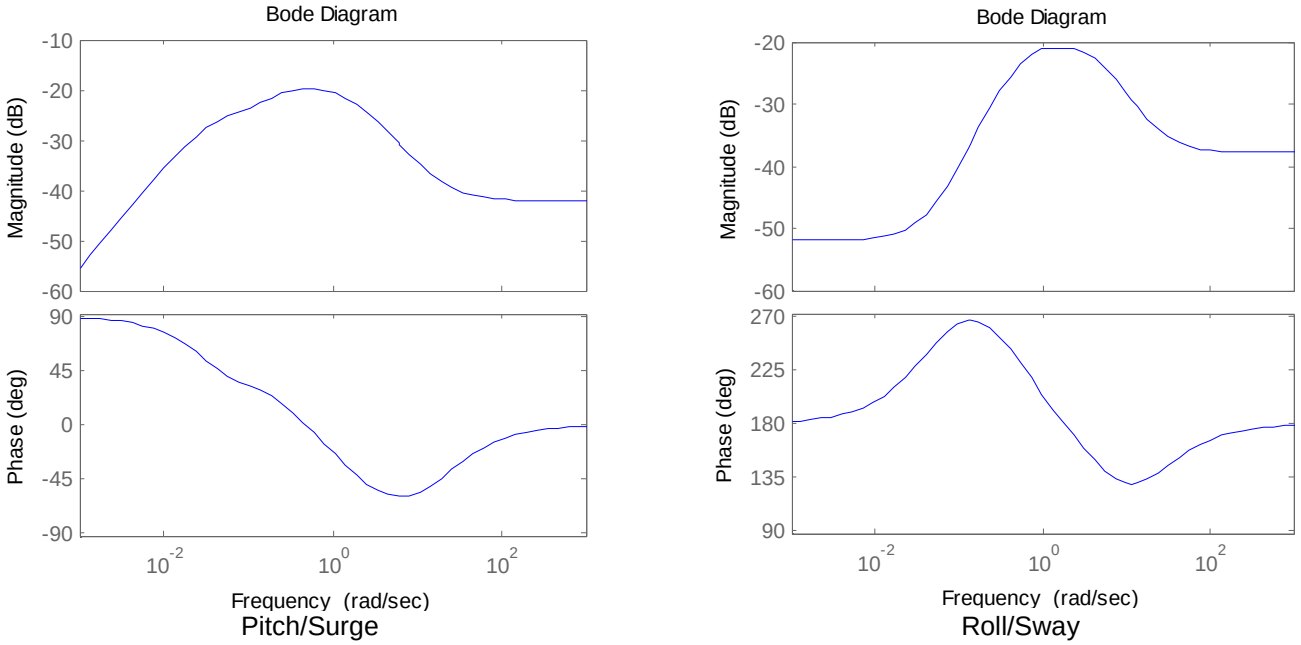


*Figure F.2: Bode plot of the coordination filter W12*

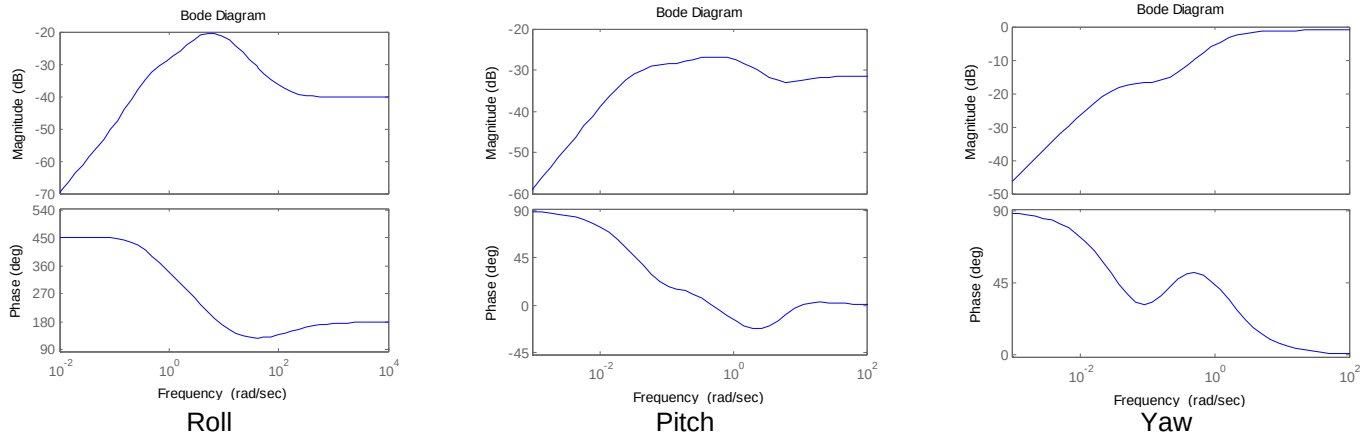


*Figure F.3: Bode plot of the rotational filter W11*

## F.9 OWF response

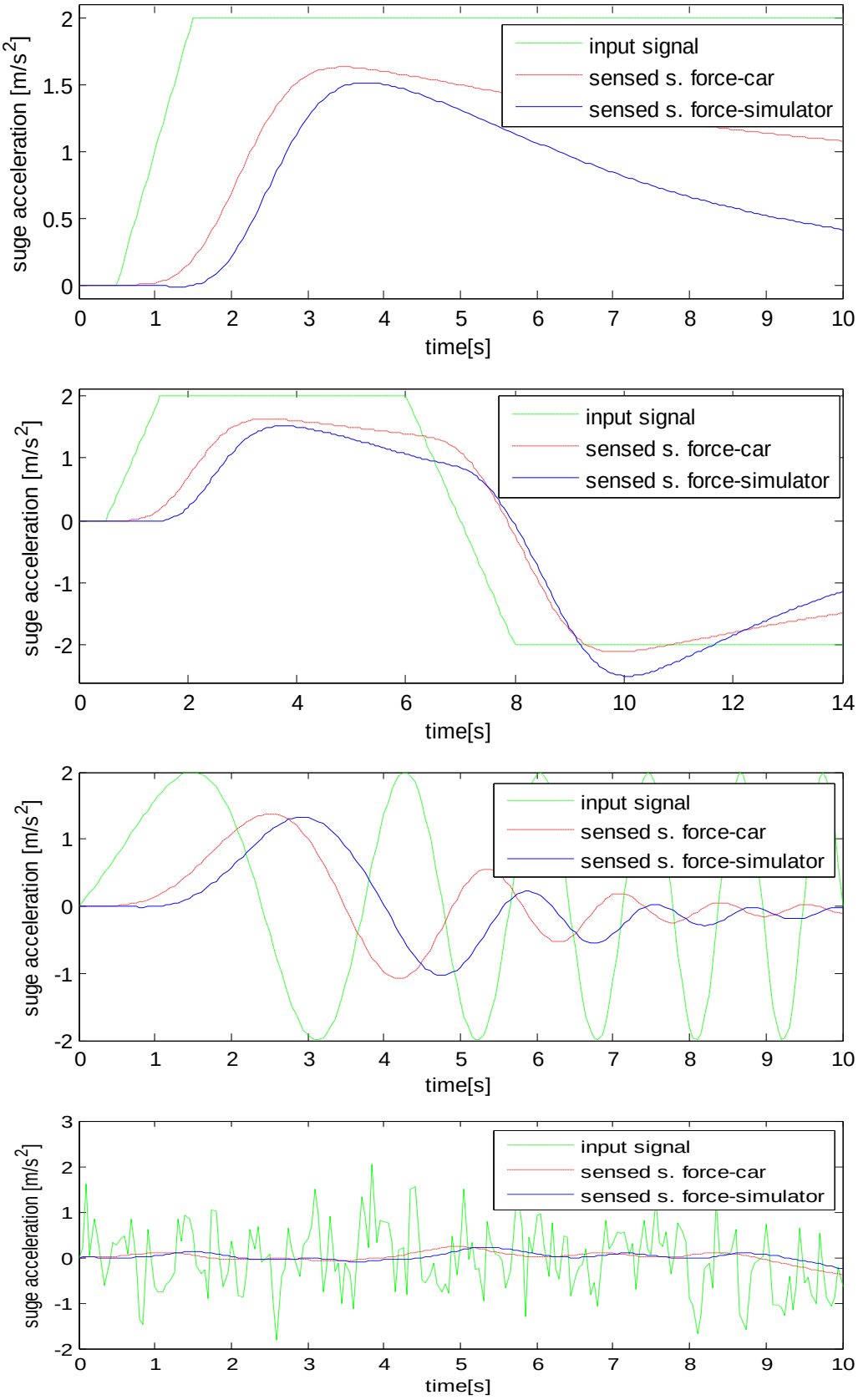


*Figure F.4: OWF response to surge acceleration a) ramp to step of $2m/s^2$, b) sudden acc. and brake, c) chirp, d) filtered white noise (note: s. force = specific force)*

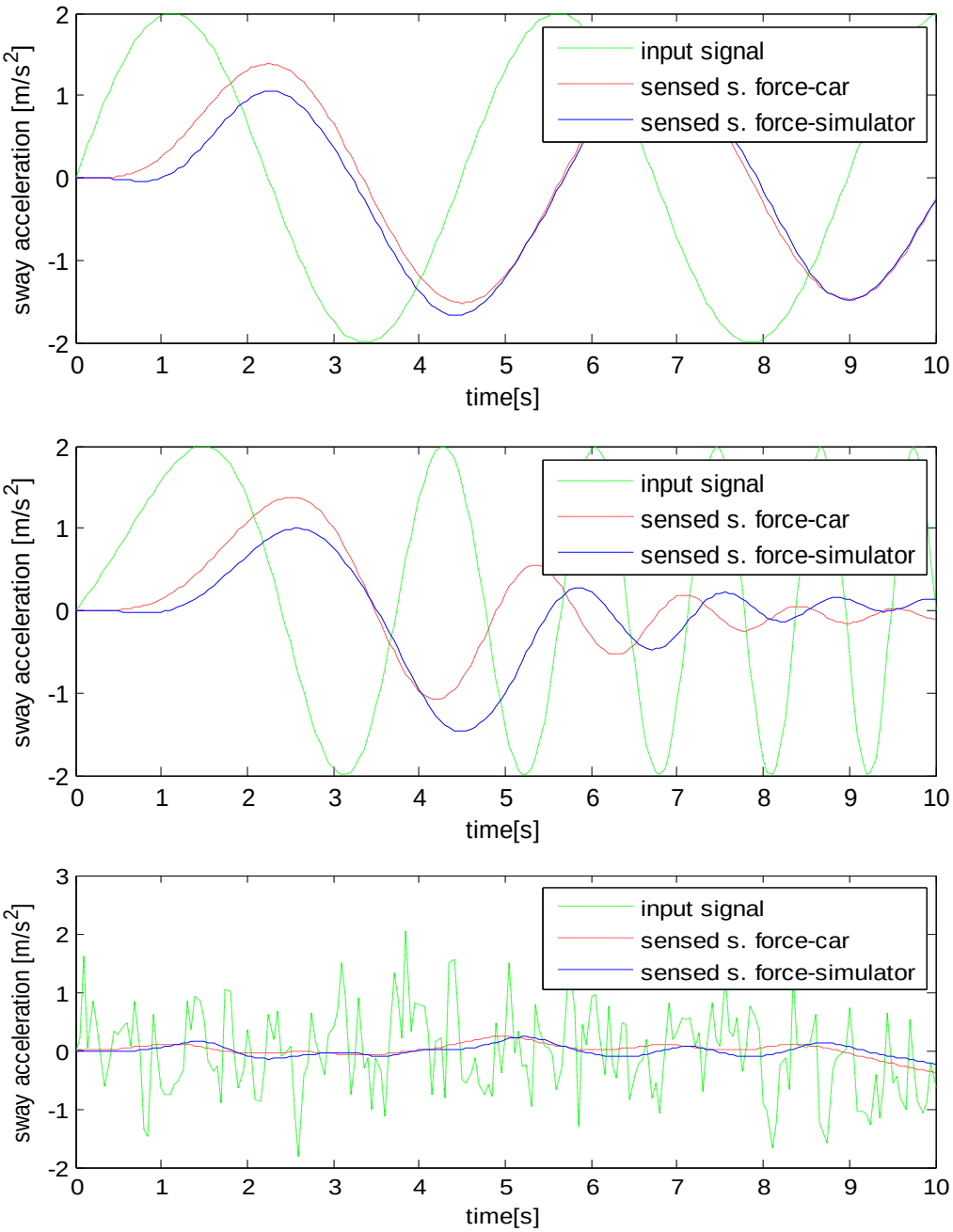


*Figure F.5: OWF response to sway acceleration a) periodic lane change, b) chirp, c) filtered white noise (note: s. force = specific force)*

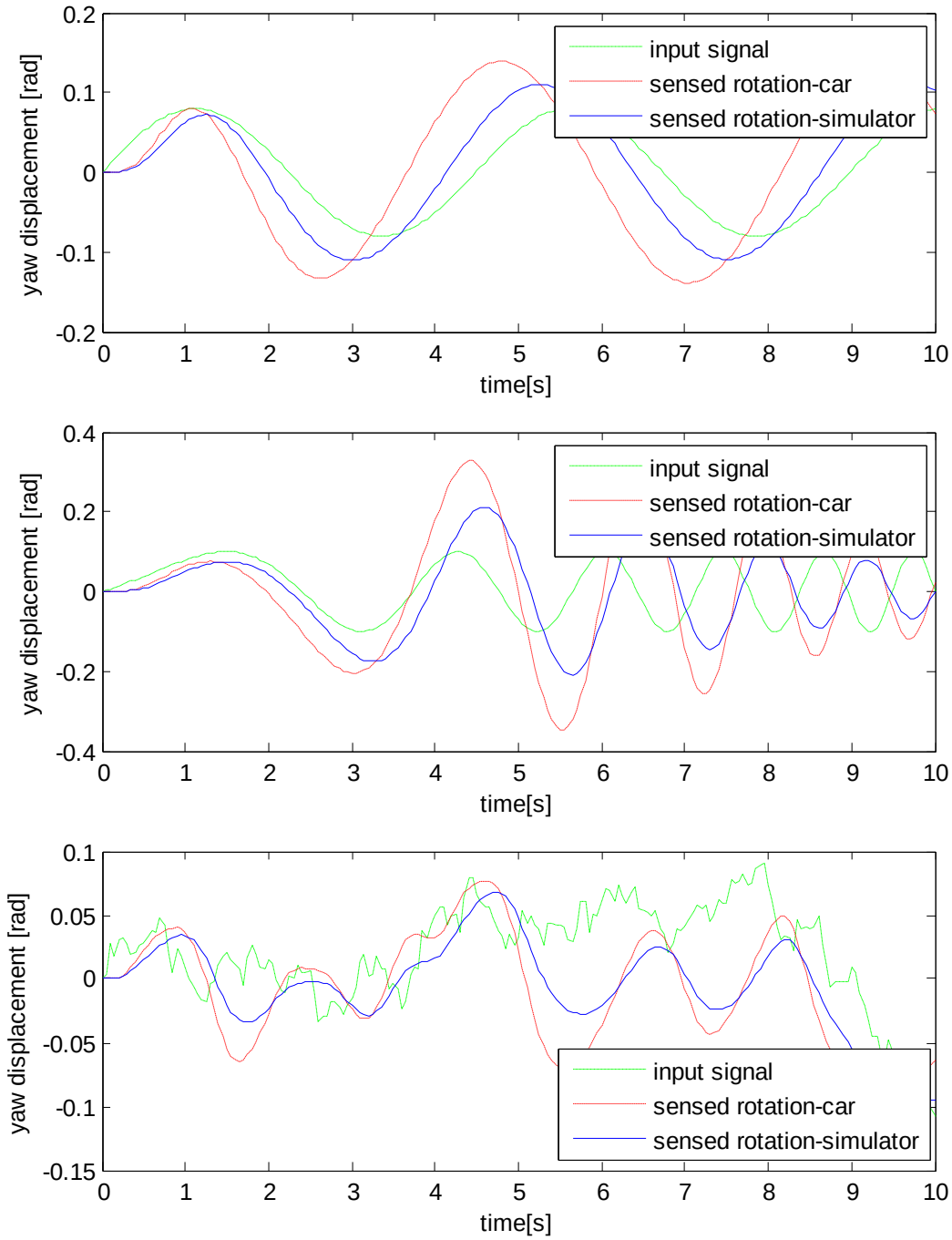


*Figure F.6: OWF response to yaw displacement*
*a) periodic lane change, b) chirp, c) filtered white noise*

## F.10 Simulink model of the OWF

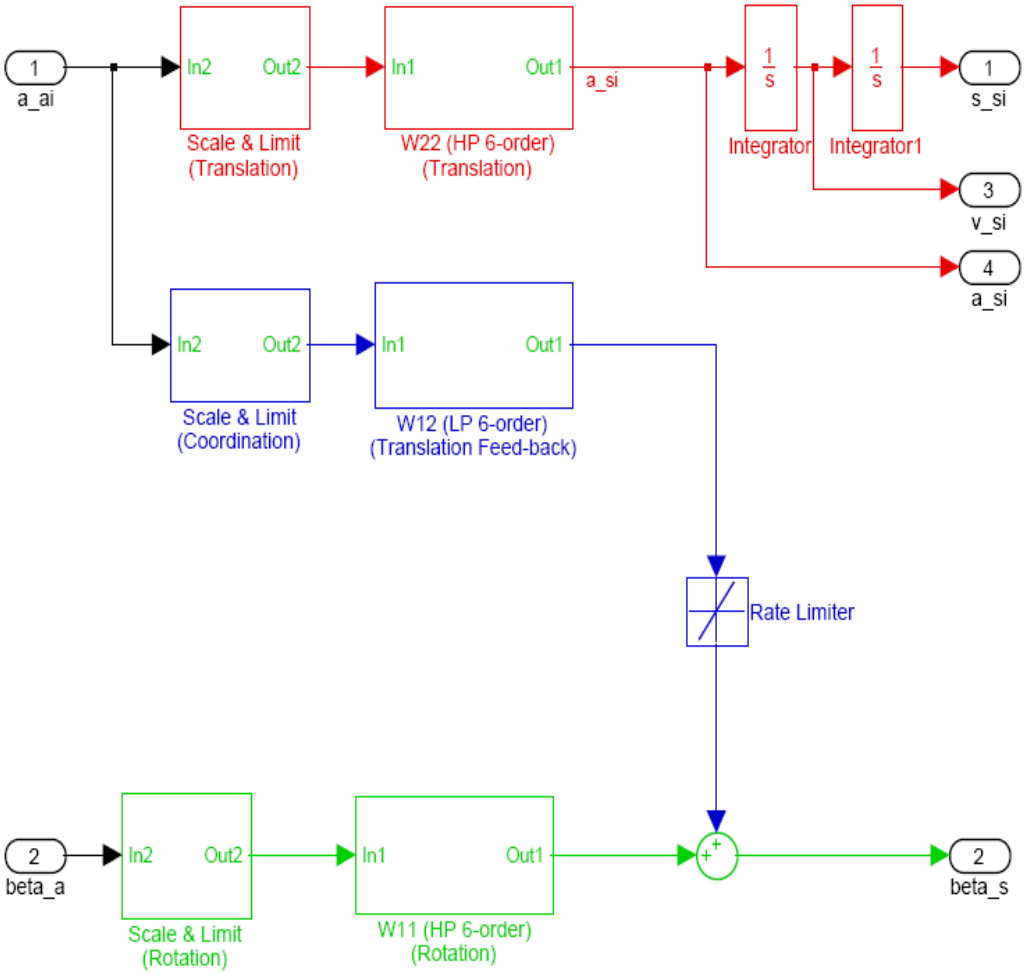


*Figure F.7: Simulink model of the OWF*

# Appendix G. Adaptive Washout Filter

## G.1 EA attributes for the AWF optimization

| | |
|---|---|
| Population size | 50 |
| Crossover probability | 0.5 |
| Type of selection | Tournament selection |
| Tournament size | 8 |
| Mutation probability | 0.1 |
| Type of mutation | Geometric creep |
| Creep rate | 0.3 |
| Number of individuals under elitism | 2 |
| Maximum generation | 500 |

*Table G.1: EA attributes for the AWF optimization*

## G.2 Values for the AWF parameters

| Parameter | Pitch/Surge | Roll/Sway | Yaw | Heave |
|---|---|---|---|---|
| $d$ | 0.0015 | 0.0824 | | 0.0125 |
| $e$ | 3.0238 | 0.1156 | | 0.3009 |
| $G_\lambda$ | 0.3009 | 1.3323 | | 0.9651 |
| $K_\lambda$ | 0.0441 | 0.0766 | | 0.1021 |
| $w_1$ | 0.4177 | 0.0182 | 8.5731 | |
| $w_2$ | 0.0954 | 0.4631 | | 0.5960 |
| $w_3$ | 0.5746 | 0.5706 | | 0.4690 |
| $G_\delta$ | 1.7096 | 0.0076 | 9.2473 | |
| $K_\delta$ | 1.2081 | 4.5414 | 0.1568 | |
| $\gamma$ | 0.0988 | -0.0801 | | |
| $RTZ_t$ | 0.5813 | 8.5796 | | 28.4149 |
| $RTZ_r$ | 2.4346 | 19.0641 | 0.0483 | |
| $\lambda_0$ | 0 | 0 | | 0 |
| $\delta_0$ | 0 | 0 | 0 | |

*Table G.2: Final values for the AWF parameters*

| **Translational channel** | **Scale** | **Higher Limit** | **Lower Limit** |
|---|---|---|---|
| Surge $[m/s^2]$ | 0.2 | 0.8 | -0.8 |
| Sway $[m/s^2]$ | 0.1 | 0.4 | -0.4 |
| Heave $[m/s^2]$ | 0.2 | 0.5 | -0.5 |
| **Coordination channel** | **Scale** | **Higher Limit** | **Lower Limit** |
| Pitch/Surge $[m/s^2]$ | 0.3 | 1.2 | -1.2 |
| Roll/Sway $[m/s^2]$ | 0.2 | 0.8 | -0.5 |
| **Rotational channel** | **Scale** | **Higher Limit** | **Lower Limit** |
| Roll *[rad]* | 1 | 0.2 | -0.2 |
| Pitch *[rad]* | 1 | 0.2 | -0.2 |
| Yaw *[rad]* | 1.8 | 0.2 | -0.2 |

*Table G.3: Final values for scale and limit*

## G.3 AWF Response

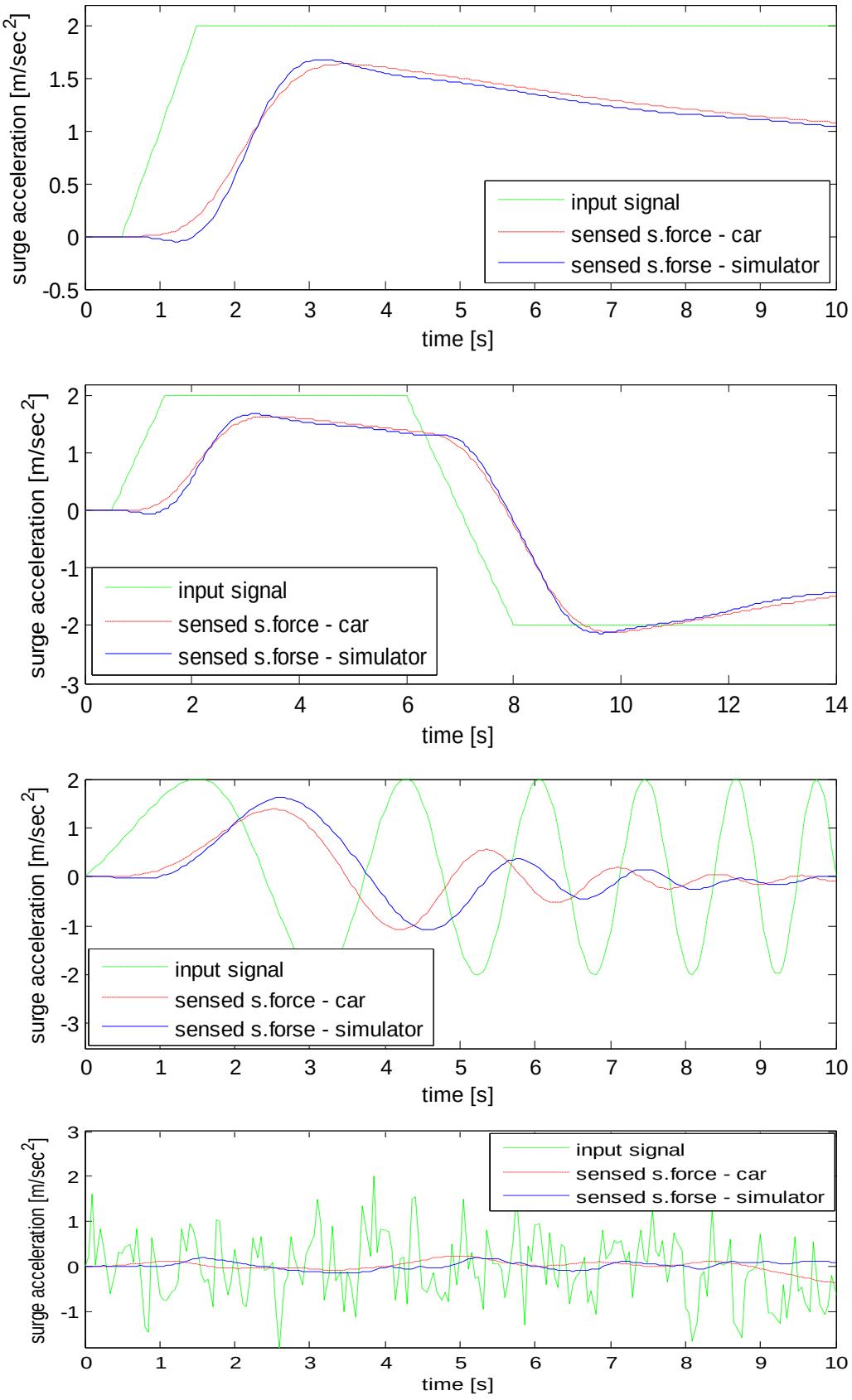


*Figure G.1: AWF response to surge acceleration a) ramp to step of $2m/s^2$, b) sudden acc. and brake, c) chirp, d) filtered white noise (note: s. force = specific force)*

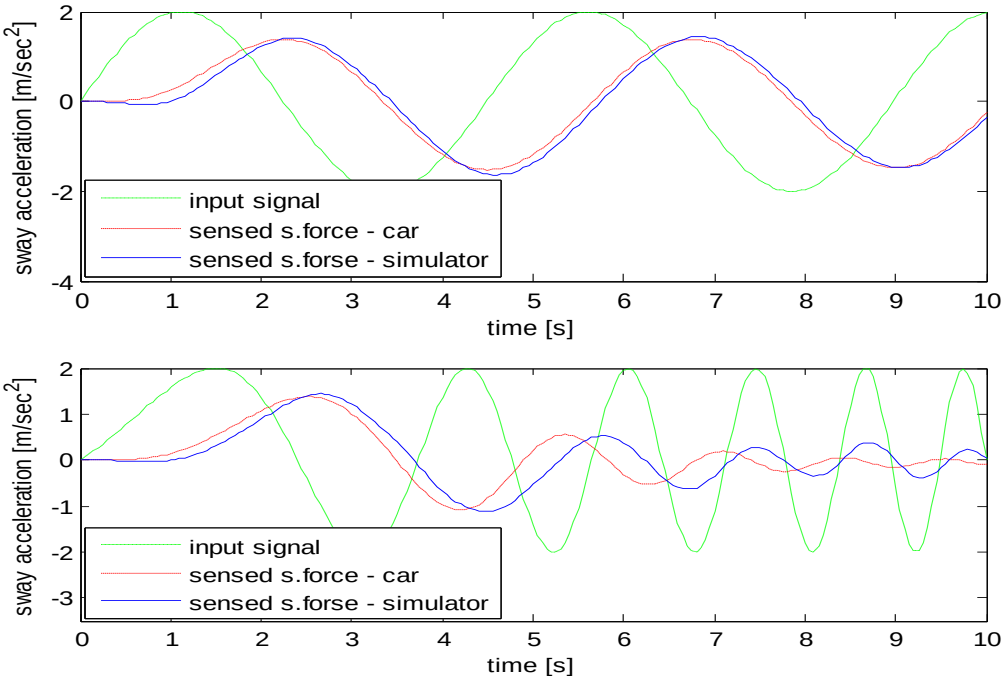


*Figure G.2: OWF response to sway acceleration a) periodic lane change, b) chirp (note: s. force = specific force)*

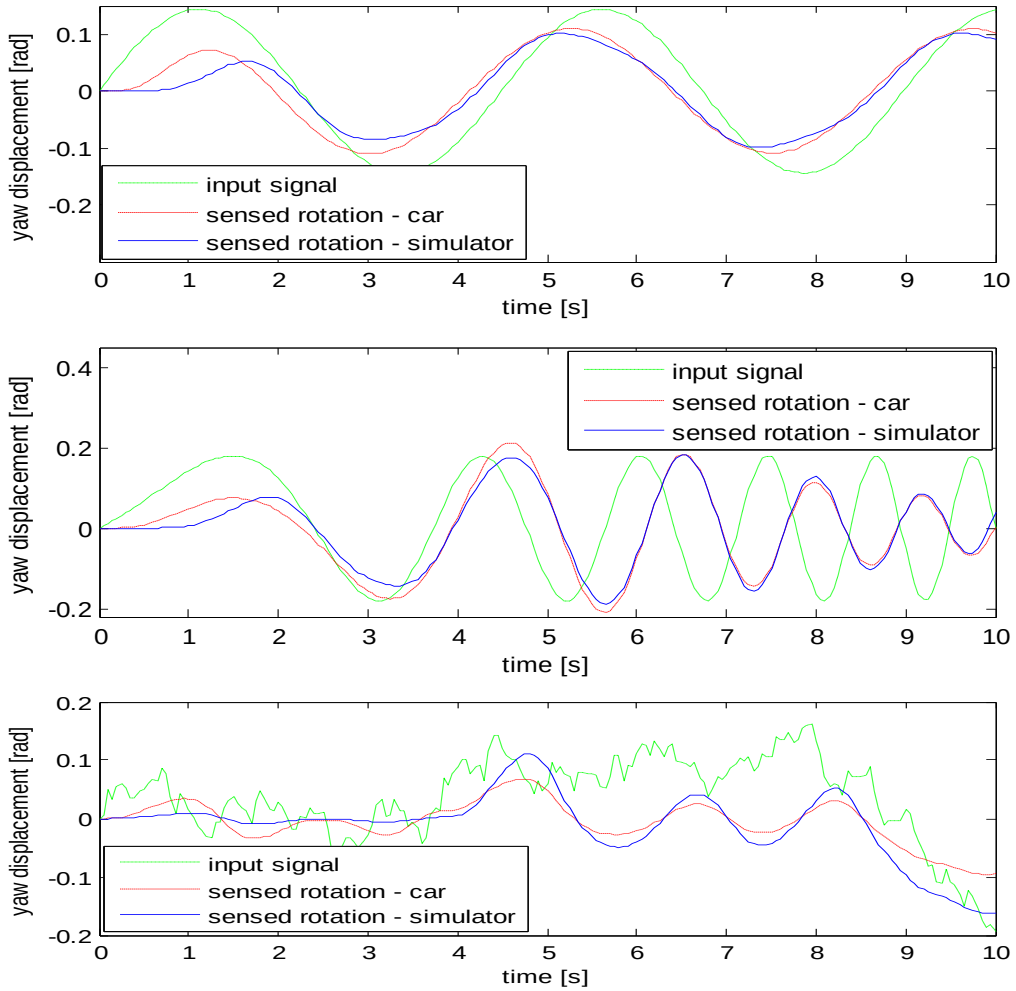


*Figure G.3: AWF response to yaw displacement a) periodic lane change, b) chirp, c) filtered white noise*

## G.4 Simulink models of the AWF

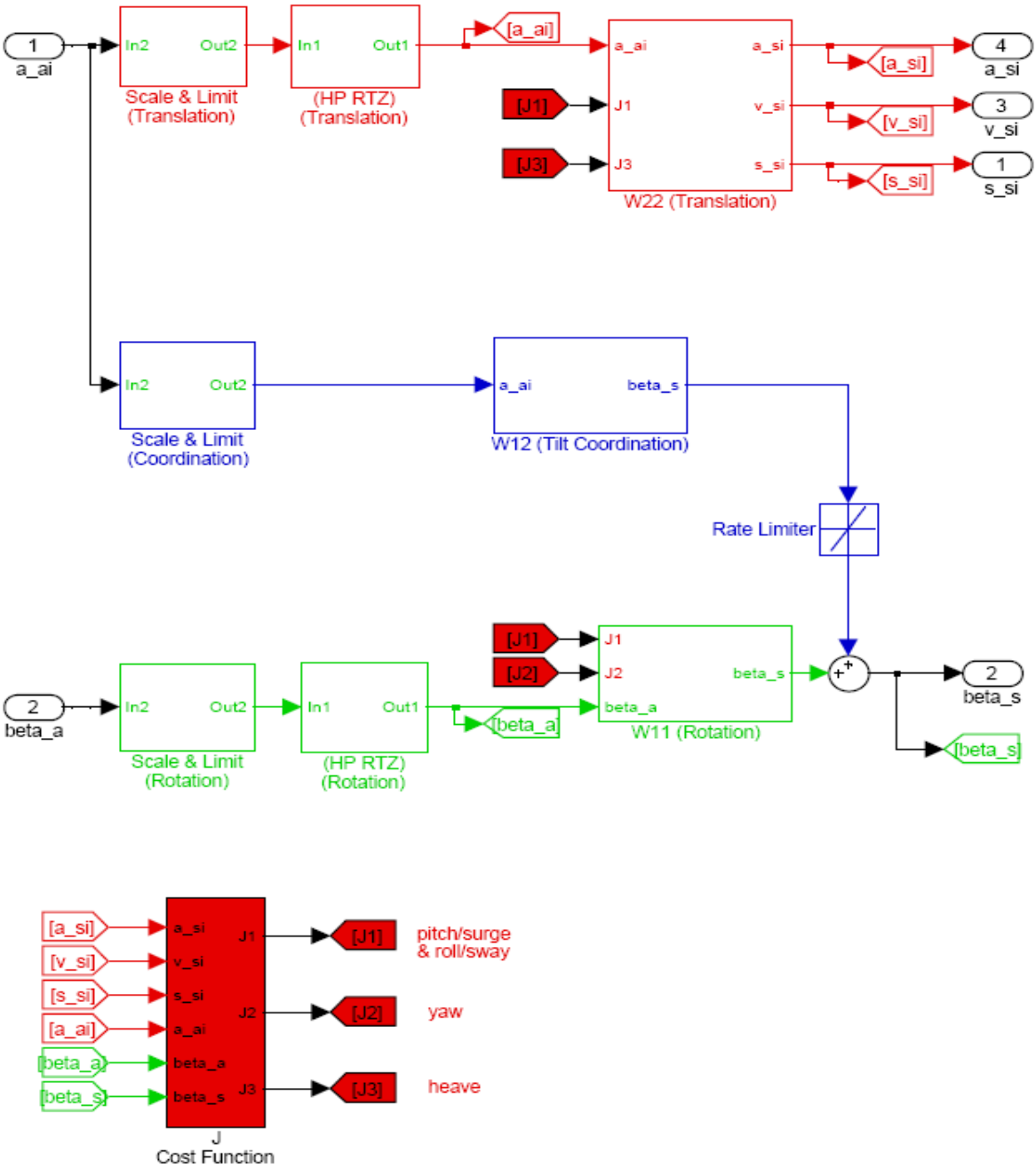


*Figure G.4: Simulink model of the AWF*

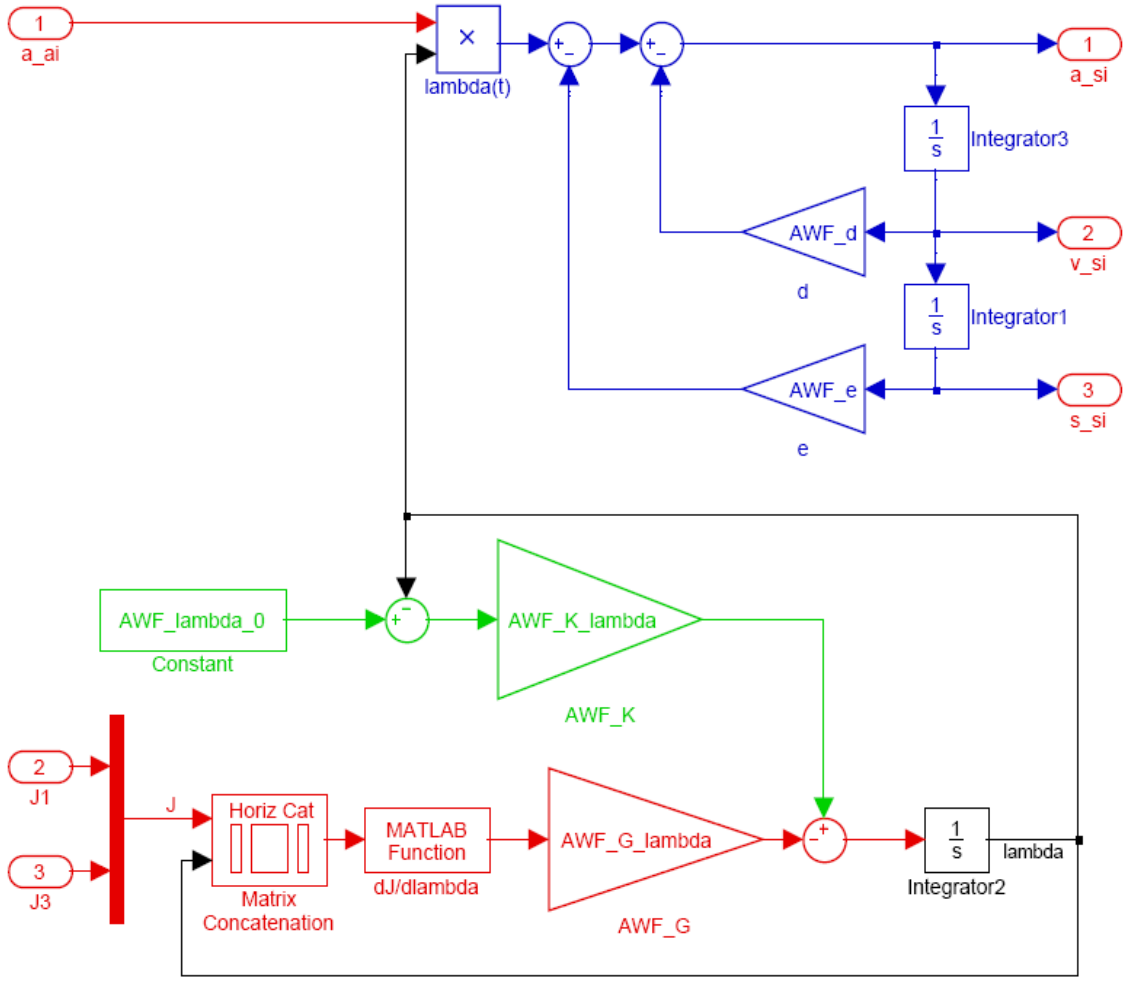


*Figure G.5: Simulink model of the AWF block W22*

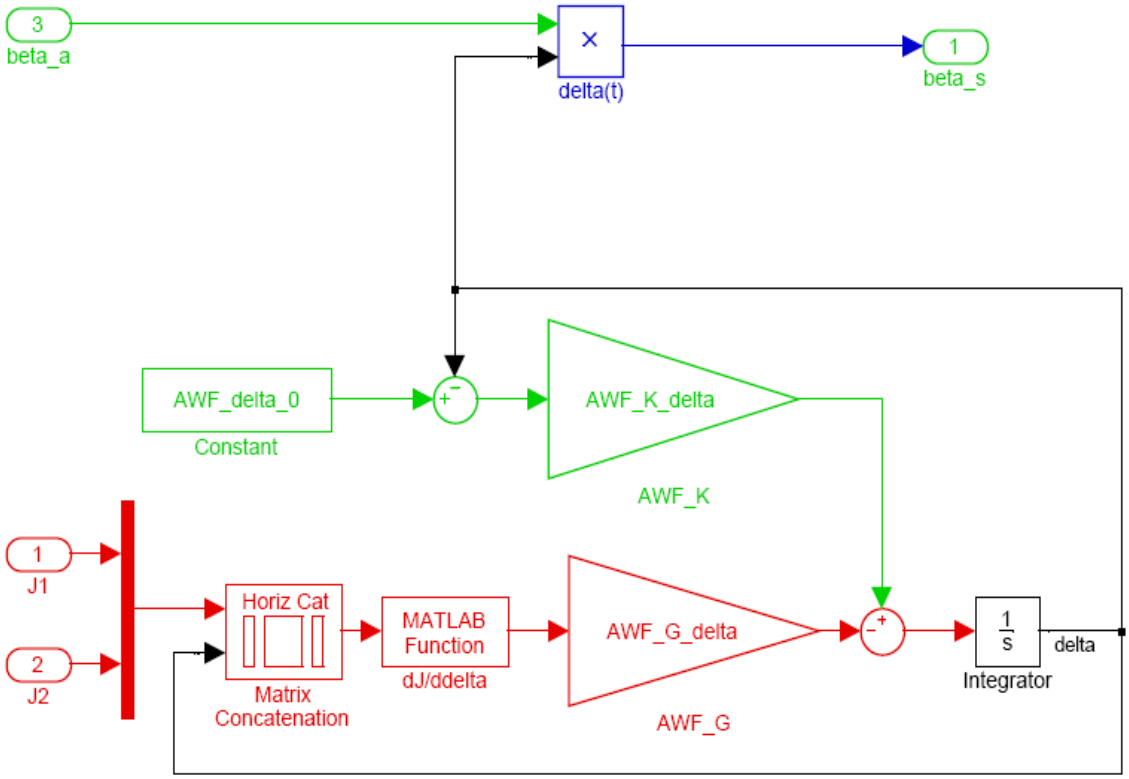


*Figure G.6: Simulink model of the AWF block W11*