UNIVERSIDAD DE LOS ANDES

Faculty of Engineering

Department of Mechanical Engineering

Masters in Mechanical Engineering

Motion System for a Dynamic Simulator

Author: Nicolás Ochoa Lleras

Advisor: Carlos Francisco Rodríguez, Ph.D.

Bogotá D. C., January 2010

TABLE OF CONTENTS

1. INTRODUCTION AND OBJECTIVES 1
2. MOTION CONTROL 4
2.1 Introduction 4
2.2 Motion System 4
2.2.1 Stewart Platform at COLIVRI Laboratory 5
2.2.2 Stewart Platform Model 6
2.3 Control System 6
2.3.1 Control Electronics at COLIVRI Laboratory 7
2.3.2 Control System Model 8
2.4 Robust Motion Controller 9
2.4.1 Modelling with Uncertainty 11
2.4.2 Robust Stability 11
2.4.3 Robust Performance 12
2.5 Results 14
2.5.1 Measurement Systems 15
2.5.2 Experimental Tests 16
2.6 Discussion 20
3. MOTION CUEING ALGORITHMS 21
3.1 Introduction and State of the Art 21
3.1.1 Classical Washout Algorithm 23
3.2 Theory of Motion Perception 24
3.2.1 Motion Cueing as an Engineering Problem 26
3.3 Proposed Approach 26
3.3.1 Motion Cueing as a Control Problem 27
3.3.2 Vestibular Sensor Model 27
3.3.3 Sensor-State Controller 28
3.4 Motion Cue Evaluation 29
3.4.1 Error Classification 30
3.4.2 Basis for Quantitative Sensor-Based Evaluation 30
3.4.3 Three Basic Elements 31
3.4.4 Two Comprehensive Indicators 32
3.5 Results 35

3.5.1 Simulation....35
3.5.2 Experimental....44
3.6 Discussion....47
4. CONCLUSIONS AND FUTURE WORK....48
4.1 Motion Control....48
4.2 Motion Cueing Algorithms....48
5. REFERENCES....50

1. INTRODUCTION AND OBJECTIVES

A dynamic simulator is a device that integrates a motion mechanism with virtual reality graphics and sound in order to recreate a real world experience. They have been used both in entertainment and training applications. In training, these are mostly aimed at aircraft pilots or vehicle drivers who must learn to read instruments and manipulate controls. In these cases, high fidelity motion recreation is not always necessary to ensure a good transfer of knowledge from the simulator to the real world (Kihl, Herring, Wolf, McVey, & Kovuru, 2006).

A special case of training arises when passengers who have no control over vehicle motion must perform certain on-board tasks. As they don't drive the vehicle, they cannot anticipate its dynamic effects and consequently their prediction of their own motion is poor. In this case, recreating the physics of motion perception is of the upmost importance, in particular when said motion interferes with on-board responsibilities, such as shooting from a moving vehicle (Rodriguez & Ochoa, Motion Simulation Based on Human Vestibular Sensors, 2008).

This work focuses on how to generate motion cues which can be reproduced with a Stewart-Gough platform, but still provide a sensory experience similar to riding on the real vehicle. The main problem being that the six-degree-of-freedom platform has limited kinematic and dynamic capabilities.

Previous works on this subject have been centred on the washout approach. This is based on using adaptative gains (Ames Research) or linear filters (Grant & Reid, 1997), (Tseng & Fong, 2000) to eliminate those aspects of the real motion which cannot be reproduced by the simulator, and to return the platform to its central position after any cue causes a displacement. The algorithm can be tuned either by defining an optimal control problem, or by using a pilot and a filter expert to adjust the algorithm parameters manually.

For this investigation, the first step taken was to understand the process of self-motion perception. This was carefully studied by (Holly & McCollum, 1996). They separated the process into two stages: a Physical one, whereby the physical motion of the head is transduced into a certain state of the body's motion sensors; and a Nervous one, by which the various nervous signals from the relevant sensors are interpreted by the brain, producing a conscious perception of motion.

Of the different body systems sensitive to motion, the vestibular system – located in the middle ear – generates the most important cues pertaining to self-motion perception (together with the eyes, but the graphics and visual system design is not part of this thesis).

In the vestibular system, the otolith organs sense linear acceleration (both gravitational and inertial, without distinction) and the semicircular canals sense rotational accelerations. Vestibular sensor response is referred to as Vestibular Afferent Firing Rate (VAFR) and it is measured in spikes/s. This is a frequency-type measurement that characterizes the nervous signals, which arise in the vestibular afferents and travel to the brain.

The behaviour of the vestibular organs has been tested experimentally by (Goldberg & Fernández, 1971) and (Goldberg & Fernández, 1976), and modelled by (Ormsby & Young, 1977), among others. A comprehensive explanation of the biomechanics and dynamics of the organs can be found in (Rabbit, Damiano, & Grant, 2004). A multi-sensory motion perception model was developed by (Borah, Young, & Curry, 1989), where information from the vestibular, visual, proprioceptive and somatosensory systems was integrated by a Kalman filter.

Whenever human perception or motion sensor models are explicitly used in simulator motion cue definition, the approach is said to be ‘human centred’. This type of methodology has produced the latest developments in motion driving mechanisms such as the non-linear adaptative filters reported in (Telban & Cardullo, 2005) and the optimal solution of (Elloumi, Bordier, & Maïzi, 2005).

This thesis focuses on developing a motion cueing system for a passenger-training simulator. In this sense, the simulator motion should be defined such that the vestibular Sensor-State of its rider, imitates that of a person on the real vehicle as closely as possible. In order to measure the quality of the Sensor-State imitation specialized error indicators were developed.

In order to implement this motion, one must consider the mechanism and control system that will be used to execute the commands. For this purpose, this work includes the modelling, simulation and testing of a Stewart-Gough Platform and its motion control system. The controller was tuned based on the Robust Performance method presented in (Doyle, Francis, & Tannenbaum, 1990) and verified on the hexapod available at COLIVRI Laboratory at the Universidad de los Andes.

Taking these aspects into account, the broad objective of this project is:

> To define a motion cueing algorithm, which computes simulator motion based on vestibular sensor response and the reference motion of the real vehicle. The resulting trajectory should reduce the specialized error indicators and be successfully reproduced with the Stewart-Gough platform available at COLIVRI Laboratory.

The proposed solution is to approach this as a control problem with the structure shown in Figure 1.1.

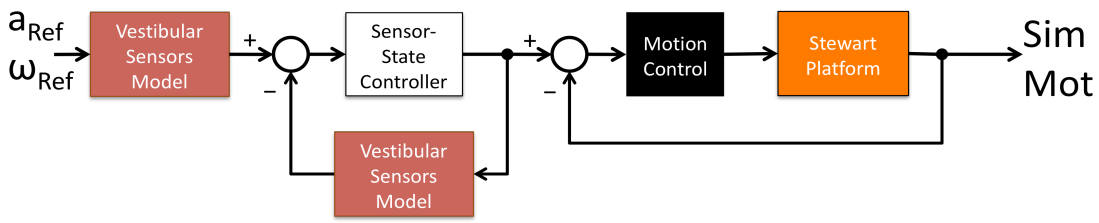


Figure 1.1 – Sensor-State and Motion controllers with split loops

In this sense, the investigation will address both the motion control problem – how to correctly reproduce a commanded motion trajectory with the six degree-of-freedom Stewart Platform – and the motion cueing problem – how to define motion commands for the Stewart Platform which provide a good recreation of the simulated motion, while keeping the platform within its kinematic and dynamic limits.

One desirable solution would be to integrate both problems by arranging the controllers in cascade, as shown in Figure 1.2. This would allow the motion cueing algorithm to adapt its commands based on the dynamic response of the controlled motion system.

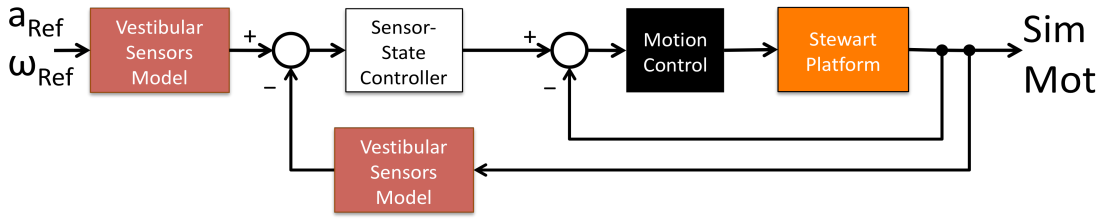


Figure 1.2 – Sensor-State and Motion controllers with cascade loops

Such an approach was tested in simulation. Its implementation on the real system, however, has various difficulties which are beyond the scope of this thesis (such as the real-time processing of motion sensors aboard the platform and the integration of the motion cueing and motion control tasks in real-time through low-level programming).

Finally, this work will show how the proposed motion cueing algorithm provides a better simulation of the reference motion than the typical washout algorithm, as well as the advantages of using a robust performance method to tune the platform's motion controller.

2. MOTION CONTROL

2.1 Introduction

When building a simulator, the upmost attention must be directed to the mechanical system that will be used to produce the desired motion. Whatever effort is put into the definition of the motion cues for the simulator, it will be irrelevant if the mechanical system – which will ultimately be responsible of executing the defined motion commands – is poorly fitted to the task.

Therefore, this aspect of the work will be addressed first.

This will be done in three stages: first, a brief description of the Stewart Platform – for any further information, the reader is directed to (Tsai, 1999), (Tseng & Fong, 2000), or the endless amounts of papers published on this mechanism; second, a brief description of the motion control system used to actuate the platform and its linear model; finally, a new approach to obtain a robust motion controller for the robot is presented.

The results at the end of the chapter show the effectiveness of the proposed method.

2.2 Motion System

The Stewart-Platform is a six DOF motion mechanism composed of two platforms and six variable length links joining them. The base platform, fixed on the ground, is joined with spherical or universal joints to the bottom part of the six legs; the top platform is joined with spherical joins to the top part of the legs. The bottom and top parts of each leg are joined with a prismatic joint, which allows them to change length. By controlling the legs' lengths, the top platform can perform linear and rotational motions in all 6 degrees-of-freedom of 3-Dimensional space. A diagram of the system is show in Figure 2.2.

The inverse kinematics problem finds the correct lengths of the six actuator for a given position and orientation of the moving platform, M, with respect to the base platform, B. This is accomplished through a simple closed-loop vector equation, where three vectors are known (position of actuator support from the centre of the corresponding platform and position vector of M relative to B), so the fourth vector (the actuator vector, from Bi to Mi) can be computed. Its norm is the necessary actuator length.

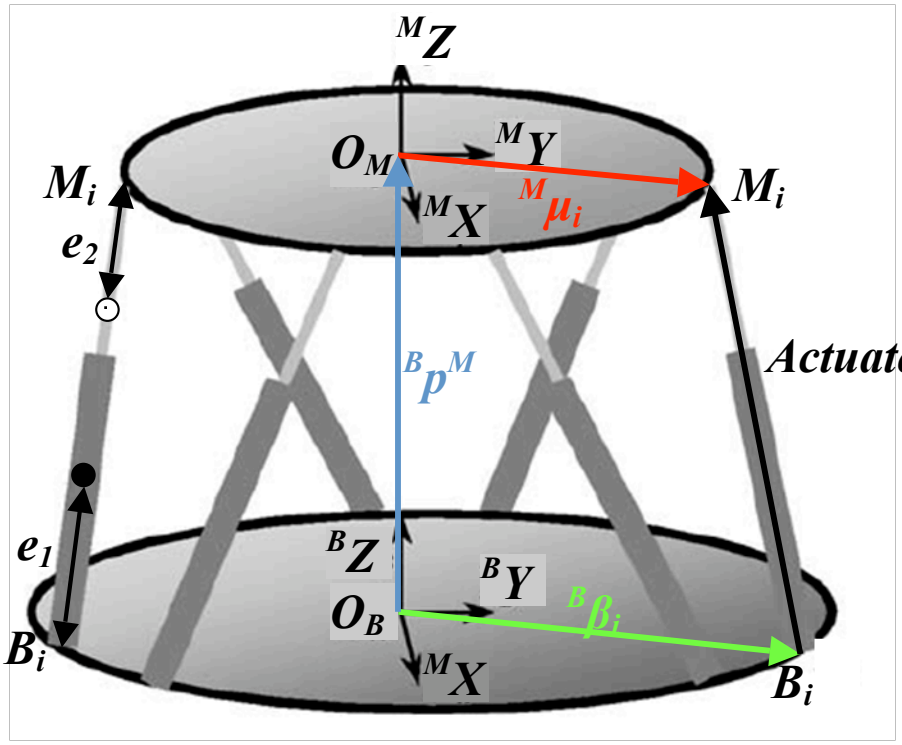


Figure 2.2 – Stewart Platform.
Taken and modified from (Tseng & Fong, 2000)

The inverse dynamics problem finds the actuator forces as a function of the kinematic variables (position, orientation and – both linear and angular – velocities and accelerations) of the moving platform. They can be modelled through Newton-Euler, Lagrange or Kane methods, and programmed in a computer to find the forces necessary to execute a given motion.

The direct kinematics (find position and orientation of M with respect to B based on actuator lengths) and dynamics (find kinematic variables of M based on actuator forces) of the platform – as is the case for all parallel manipulators – are too complicated to be solved analytically (except for very specific types of geometric configurations), and must be solved numerically through iterative methods.

2.2.1 Stewart Platform at COLIVRI Laboratory

The mechanical device that will be used to test the algorithms in the coming semester has a base platform radius of 0.402 m and a moving platform radius of 0.265 m. Actuator supports at the base are spaced in couples every 120° with a +/- 10° offset, while the supports at the top are also spaced every 120° but with no offset, so that the couples share a single spherical joint at the top.

It was constructed with electromechanical actuators. These actuators have a rotary servomotor of 400 kW at the lower leg, connected to a screw of lead 0.4 in per

revolution and a total of 12 in of stroke (they vary their length from 0.936 m to 1.253 m) at the top leg. By introducing a controlled current to the servomotor, it will produce a torque that turns the screw in or out of the bottom leg, thus altering its length. By controlling the lengths of the six actuators, one can produce almost any movement in 3D space within the cinematic limits of the actuators (limited by their stroke and the servomotor's rpm – 5000 rpm maximum (Yaskawa Electric America, Inc, 2009)).

The actuators and servomotors also have dynamic limits, which must be taken into account when programming motions. The servomotors can provide up to 1.27 Nm of continuous torque (3.82 Nm for peaks) (Yaskawa Electric America, Inc, 2009), while the actuators tolerate up to 854 N of continuous force (1676 N for peaks) (Exlar Corporation, 2008).

2.2.2 Stewart Platform Model

A model of the platform was constructed in Simulink, using the SimMechanics toolbox. The process of building this model is thoroughly explained in (Smith & Wendlandt, 2002). The system parameters – such as geometry, element masses and inertias – where defined so that the model resembles the physical system at COLIVRI Laboratory.

A small difference is that the model is actuated with a force (introduced in the prismatic joint which joins the lower and upper leg), instead of a rotary servomotor. None-the-less, the motors' torque can be easily converted to the resulting force by using the constant relationship between angular and linear velocity of the screw. This relationship is given by the screw's lead.

Masses and inertias of the leg elements where estimated using information from the respective catalogues of the actuators (Exlar Corporation, 2008) and the servomotors (Yaskawa Electric America, Inc, 2009). The mass and inertia of the top platform were estimated by using a CAD model of everything that's mounted on the moving platform.

2.3 Control System

A motion control system for the Stewart Platform would control the torque (or, the angular velocity) that the servomotors exert on the actuator so that the platform follows a designated sequence of points or velocities in time. One could also use the platform to apply a controlled force, but this case does not apply for simulators.

The controller would use feedback from incremental or absolute encoders mounted on the motors to determine the actuator's kinematic variables (position, velocity and acceleration). Its output is usually an analogue voltage signal, which is most

commonly determined by a PID control law. Other typical laws for motion include pseudo-derivative and feed-forward methods.

Such signal would travel to an amplifier, which would output a current proportional to the input voltage. This current would travel to the servomotor, thus controlling its torque (or angular velocity).

2.3.1 Control Electronics at COLIVRI Laboratory

The motion control system to be used is comprised of a Yaskawa SMC-2000 multi-axis control card and six Legend04 amplifiers. The control card is connected to a PC, which uses YTerm software (provided by the manufacturer) to communicate and control the SMC-2000.

The six actuators can be controlled simultaneously, but the controller still handles every actuator independently. What this means is that one cannot command directly a desired platform position and orientation (as the control card does not know the mechanic configuration of the actuators being controlled). Instead, the commands are the individual actuator positions or velocities as a function of time.

For this reason, the inverse kinematics function is essential in order to translate a desired platform position and orientation into the commands that the control card can use to control the actuator's motion.

Motion feedback is provided by incremental encoders of 2096 pulses per revolution (ppr) with cuadrature 4, thus generating 8192 ppr. The servo's rotary motion translates into linear displacement of the actuator through the screw lead.

The actual control algorithm is a PID discrete (digital) filter with a sample rate of 1 kHz. The manufacturer provides equations to calculate the appropriate gains for an equivalent continuous PID filter (Yaskawa Electric America, Inc, 2004). The controller's digital output is converted to an analogue voltage signal through a Digital-to-Analogue-Converter of constant gain of 20/65536 Volts. The voltage travels to the amplifier, which generates a current so as to maintain a proportional relationship between the input voltage and the servomotor's output torque.

In summary, when the commands are sent to the SMC-2000 card, it produces the necessary voltage output to move the actuator to the desired angular position (measured in encoder pulses), which consequently changes the actuator length and allows the platform to move in 3D space.

2.3.2 Control System Model

Since the controller handles each of the six axes independently, a simple controller-plant model was developed for a single actuator. The idea behind this model is that it can be used to tune the controller for one actuator – which facilitates the controller tuning process. The PID gains found to be appropriate through this method should be good for all six actuators. If all actuators are correctly tuned, then the Stewart Platform, as a whole, will be reasonably well controlled. We also intend to use this tuning process to tune the PID controller that will drive the Stewart Platform Simulink model. This subject of controller tuning is more deeply addressed in the next chapter.

Figure 2.3 shows a schematic of how we can set about modelling the controller-actuator system. The zoh (zero-order-hold), DAC (Digital-to-Analogue Converter), amplifier and encoder can me modelled as constant gains, using data provided by the manufacturer:

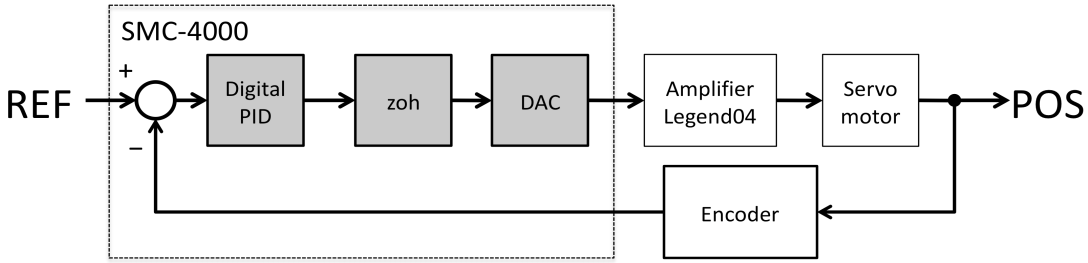


Figure 2.3 – Motion control system model for single actuator

Zoh: *Kz = 1*
DAC: K_{DAC} *= 20/65536 Volts*
Amplifier: *Ka = 0.382 Amperes /Volt*
Encoder: *Ke = 4 * (2096 pulses/rev) * (1/2π rev/rad)*

The digital PID filter must be implemented in MATLAB with the correct sample time of 0.001 s. Its input is position error in pulses, and its output is a (digital) number between +/- 65536. The zero-order hold converts this to a continuous signal and the DAC converts it to an analogue voltage between +/- 10 Volts. The filter has the following transfer function:

$$PID = \frac{K \cdot (z - A)}{z} + \frac{C \cdot z}{z - 1}$$

where the *K*, *A* and *C* gains correspond to the proportional, integral and derivative actions of the controller, respectively. This transfer function can be replaced with its continuous equivalent:

$$PID = \frac{D \cdot s^2 + P \cdot s + I}{s}$$

where *P*, *I* and *D* correspond to the proportional, integral and derivative actions of the controller, respectively. These can be computed from:

$$P = K(1 - A) \quad D = Ts \cdot K \cdot A \quad I = C/Ts$$

where Ts is the controller sample time, 0.001 s. All of the previous models and constants are in accordance with the controller's manual (Yaskawa Electric America, Inc, 2004).

The servomotor model uses the amplifier current as input, and the servo's angular position (in radians) as the output. It must include the torque constant – K_T, by which we assume that the input current is proportional to the applied torque – as well as the inertia of the object that the servo is driving. The model also included an estimation of the viscous friction parameter, c_{fr}:

$$Servo = K_T \frac{1}{s \cdot \left(J \cdot s + c_{fr}\right)}$$

The inertia, *J*, was estimated by adding the actuator inertia – calculated according to (Exlar Corporation, 2008) – and the equivalent inertia of the load that the actuator must move. This equivalent inertia is defined such that, when it rotates at a given angular velocity, it accumulates the same amount of kinetic energy as the mass moving with the corresponding linear velocity. Therefore the mass and inertia are related through the square of the screw lead. The friction, c_{fr}, was chosen so that, when the servo has no applied load, at the rated current of 2.8 A the angular velocity remains constant at the rated speed of 3000 rpm.

2.4 Robust Motion Controller

It was decided that a model for a single actuator would be used in order to tune a PID controller that allows the actuator to follow a certain trajectory (sequence of positions in time) while carrying a certain load. This load would be represented in the model in the inertia parameter of the servo's transfer function. The obtained values of the PID gains would then be implemented on all axes, both on the Yaskawa control card and the Stewart Platform model in Simulink.

In order to facilitate this application to both models (single actuator and Stewart Platform) and to the real control card, the controller transfer function was defined as

$$C(s) = K_{DAC} \cdot K_A \cdot K_T \cdot \left(K_P + K_D \cdot s\right)$$

Where the input is the position error in pulses, and the output is the servomotor torque in Nm. This torque can be easily converted to the force necessary for PID control of the platform Simulink model: F = 2π/Lead * T. An integral gain was not considered since we are more interested in following trajectories, rather than final stationary positions. This also simplifies the controller tuning process.

The nominal plant to be controlled is therefore defined as:

$$P_0(s) = \frac{K_E}{s \cdot \left(J_0 \cdot s + c_{fr}\right)}$$

Where its input is servomotor applied torque, and its output is the servo's angular position in pulses. The inertia is computed as:

$$J_0 = J_{SERVO} + J_{SCREW} + m_0 \cdot \left(\frac{Lead}{2\pi}\right)^2$$

Where m_0 is the nominal mass of the load attached to the actuator, and Lead is the screw lead in meters, 10.16 x 10^{-3} m.

This mass parameter, however, is not constant for two reasons: different passengers have different masses, so the overall load carried by the simulator varies; and, as the platform moves the forces that each actuator applies on the moving platform also vary.

We want the controller to perform well under any possible circumstance. This includes heavy and light passengers aboard the simulator, as well as extreme positions in which a single actuator may have a very high or very low load.

Using an inverse dynamics algorithm for the Stewart platform with the corresponding geometric and mass properties, a non-exhaustive search of extreme load conditions within the kinematic workspace was carried out.

It was estimated that the load carried by a single actuator could vary between 10 kg and 80 kg. This is also consistent with the maximum continuous force that the actuators can tolerate (854 N). Therefore, our plant model has parametric uncertainty.

2.4.1 Modelling with Uncertainty

This parametric uncertainty means that we must tune our controller to perform well, not only for a single value of the mass, but also for a whole range of them. In this sense, the controller must work well for a family of plants. This family of plants can be defined as:

$$P(s) = \frac{K_E}{s \cdot \left((J_0 + \Delta J) \cdot s + c_{fr} \right)} \quad where \quad \Delta J = \Delta m \cdot \left(\frac{Lead}{2\pi} \right)^2 \quad and \quad \Delta m \in [-35, 35]$$

This can be modelled as a multiplicative uncertainty of the form:

$$P(s) = P_0(s) \cdot \left(1 + \Delta_P(s) \cdot W_P(s) \right)$$

$$P(s) = \frac{K_E}{s \cdot \left(J_0 \cdot s + c_{fr} \right)} \cdot \left(1 + (-1) \cdot \left(\frac{\Delta J}{(J_0 + \Delta J)s + c_{fr}} \right) \right)$$

Where, the uncertainty weight function,

$$W_P(s) = \left(\frac{\Delta J_{MAX}}{(J_0 + \Delta J_{MAX})s + c_{fr}} \right)$$

is the upper bound of the model uncertainty. The value ΔJ_{MAX} occurs when Δm=35kg. J_0 is calculated for the nominal mass, m_0 = 45 kg.

In this sense, the controller must exhibit a good behaviour for all the possible plants P(s), instead of only the nominal plant $P_0(s)$. In the next two sections we define what 'good behaviour' means for the Stewart Platform. The basic theory for this uncertainty analysis, as well as the tools presented in the next subsections can be found in (Doyle, Francis, & Tannenbaum, 1990).

2.4.2 Robust Stability

The first aspect of 'good behaviour' is that the feedback system is internally stable. This can be proved if the following condition is met:

$$\left\| W_P(j\omega) \cdot T_0(j\omega) \right\| < 1 \quad \forall \quad \omega, \qquad \text{or,}$$

$$\left\| W_P(j\omega) \cdot \frac{C(j\omega) \cdot P_0(j\omega)}{1 + C(j\omega) \cdot P_0(j\omega)} \right\| < 1 \quad \forall \quad \omega$$

The double bars denote the system norm (or gain), j is the imaginary number operator, ω is the frequency and $T_0(s)$ is the closed-loop system transfer function. What this condition is imposing is that all of the possible feedback system transfer

functions be stable. Since Wp(s) and $P_0(s)$ are fixed, we must choose the controller parameters K_P and K_D so that the condition is met.

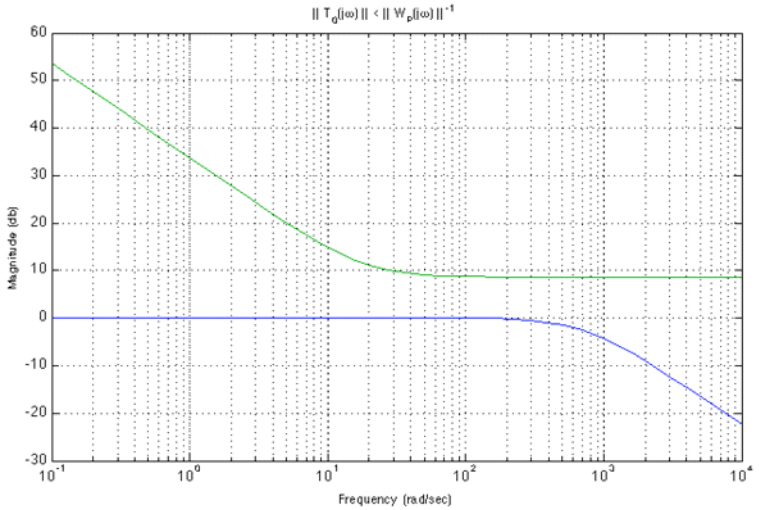


Figure 2.4 – Bode plot proving robust stability

The Bode plot shows how this condition is met for all the frequency range with the constants K_P = 12.9 and K_D = 0.29. The blue line corresponds to the Gain vs. Frequency plot of $T_0(j\omega)$, which is always below the green line which is the gain of $W_P(j\omega)^{-1}$ Therefore, from Figure 2.4 we can conclude that the plant complies with the first condition: that the feedback system is internally stable for all the possible plants P(s).

2.4.3 Robust Performance

Since the controller-plant system has robust stability, we can now specify the performance objective we want to meet with the controller. In particular, we want to asymptotically track reference trajectories within a certain range of frequencies with a small tracking error.

By a thorough analysis of the vestibular system models reported in (Borah, Young, & Curry, 1989) and (Rabbit, Damiano, & Grant, 2004) and the experimental results show in (Goldberg & Fernández, Physiology of Peripheral Neurons Innervating Otolith Organs of the Squirrel Monkey. I, II & III., 1976) and (Goldberg & Fernández, Physiology of Peripheral Neurons Innervating Semicircular Canals of the Squirrel Monkey. I, II & III., 1971), it was possible to conclude that the sensors respond to motion signals in the range of 0 rad/s to about 12.5 rad/s. Therefore, the performance weight function is defined as:

$$W_R(s) = \begin{cases} 1/\gamma & for \quad 0 \le \omega \le 12.5 \\ 0 & for \quad 12.5 < \omega \end{cases}$$

And the robust performance condition is defined as

$$\left\|W_R(j\omega)\cdot S_0(j\omega)\right\| + \left\|W_P(j\omega)\cdot T_0(j\omega)\right\| < 1 \quad \forall \quad \omega$$

Where S_0 (s) is the sensitivity transfer function, $S_0(s) = 1 - T_0$ (s).

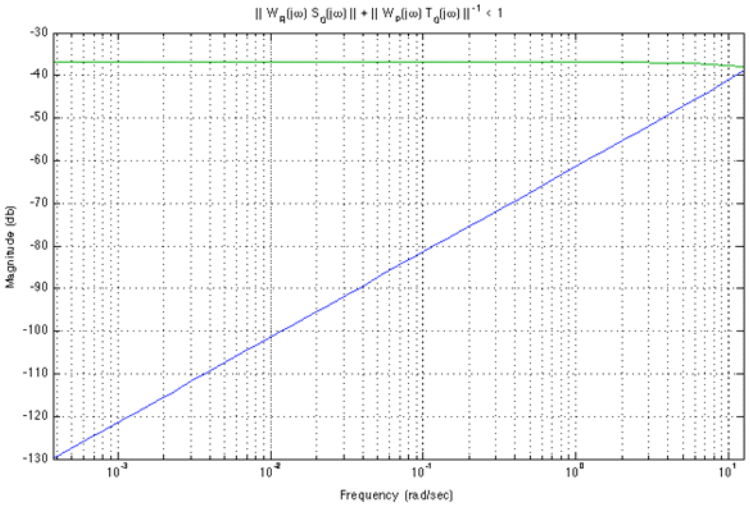
The parameter γ in the $W_R(s)$ function can be selected by the control designer. Its significance is that this value will be the maximum energy of the tracking error, as a percentage of the input signal energy:

$$\gamma \geq \|e\|_2 / \|u\|_2$$

Therefore, if we want to track an input signal u(t), with unitary energy (or 2-norm), the energy (or 2-norm) of the error signal, e(t) will be below γ. For this reason, γ should be as small as possible.

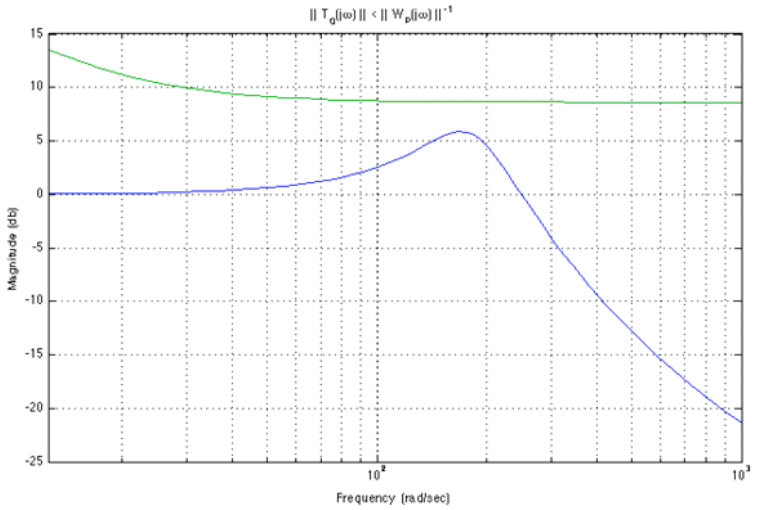
For the controller parameters given above, γ was chosen as 1.4 %. The bode plots of Figure 2.5 show how the robust performance condition is met for the frequency range from 0 rad/s to 12.5 rad/s (9a, where $W_R(s) = 1/\gamma$) and from 12.5 rad/s onwards (9b, where $W_R(s) = 0$). From the graph, we can tall that the gain of the sensitivity function (in blue in a) is always below the maximum permitted value, plotted in green. The same happens for the closed-loop transfer function in graph 9 (b).

What this all comes down to, is that this PD controller will be able to follow trajectories accurately for frequencies between 0 and 12.5 rad/s, as is proved by the very small upper bound on the error signal energy of 1.4%. This is true for a wide range of loading conditions, from 10 kg to 80 kg of equivalent mass.



(a)

Figure 2.5a – Bode plots for robust performance



(b)

Figure 2.5b – Bode plots for robust performance

This should facilitate the extrapolation of the single-actuator controller to the whole Stewart Platform, where loading conditions are constantly changing as the platform moves within its workspace. With this procedure, we have theoretically proved that the controller will have a good behaviour under the described variations in individual actuator loading masses.

The PD constants can be converted to the units used by the control card software (KP = 12.9, KD = 290) in order to apply this controller to the real control system used on the Stewart Platform at the Lab.

2.5 Results

Once the controller gains were obtained, the same model was used in simulation to verify that the model behaves as expected. This means that the error signal's energy is bounded by the inverse of γ under the maximum and minimum loading conditions considered, when the commanded motion is within the defined frequency range.

This exercise is simply another way – different from simply looking at the Bode plots in Figures 2.4 and 2.5 – of verifying that the specified performance objectives were met, at least for the proposed model. For this reason, there is no need to reproduce those results here.

On the other hand, experimental results are vital to validate that the proposed robust control approach not only works for the individual actuators, but also for the group of them interacting on the Stewart Platform to execute a commanded motion in terms of platform position and orientation. For this purpose, three different measuring systems

were employed. Such systems as well as relevant results are shown in the following sub-sections.

2.5.1 Measurement Systems

Encoders
For direct measurements of the actuator's state, the relative encoders (with 8192 ppr) provide very reliable and precise information. The motion commands for the servos are expressed in terms of pulses, making the encoders a reliable sensor to measure the individual actuator's response at the system level.

Direct Kinematics
Computing the direct kinematics for parallel robots in closed form is practically impossible due to the highly non-linear nature of the equations (Tsai, 1999). It is, however, very simple to estimate the position and orientation of the platform with information from the actuator's lengths by using an iterative algorithm. The dynamic series shown in Figure 2.6 implements this idea.

By providing a sensible guess for the initial position-orientation vector, x_0, (for example, the commanded value) and the measured actuator lengths, L, the algorithm uses the inverse Jacobian and inverse kinematics (both of which are simple calculations for the Stewart platform) to produce a better estimation, x_{i+1}, of the platform's state (Almonacid, Saltaren, Aracil, & Reinoso, 2003). The iteration stops when the difference between the measured and estimated actuator's lengths is below the convergence radius, CR: $|| l_{i+1} - L|| < CR$. Testing the algorithm revealed that it is generally possible to obtain a CR in the order of micrometers within 9 iterations.

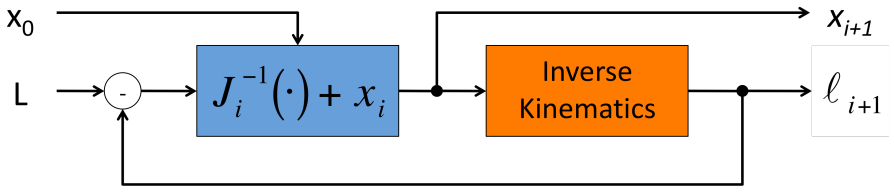


Figure 2.6 – Direct kinematics iterative algorithm

Inertial Mounted Units
Two different Sparkfun inertial mounted units (IMU) were placed on the moving platform. The SEN-00741 is a low-cost, 5 degree-of-freedom sensor equipped with 3 orthogonal accelerometers and two orthogonal rate-gyroscope. It has a measuring range of ± *3 g* for acceleration and ± *500 °/s* for angular velocity; the nominal sensitivities are of *300 mV/g* and *2 mV/ °/s* respectively.

The Sparkfun Barnacle is 6 degree-of-freedom IMU with measuring ranges of ± *4 g* and ± *75 °/s* and nominal sensitivities of *550 mV/g* and *900 mV/ °/s*, for acceleration and angular velocity respectively.

The Barnacle IMU seems to be more adequate for the ranges of motion to be measured on the simulator. However, this sensor was found to be sensitive to electromagnetic interference from the platform's servos. Therefore its signal became very noisy when the platform was in motion. For this reason, these measurements were backed up with the cheaper SEN system.

Both units had to be calibrated for bias and sensitivity. This was done by performing static tilts at precisely controlled angles on a milling machine.

2.5.2 Experimental Tests

These three sets of results will show how the PD controller defined through the robust control method produces a good behaviour of both the individual actuators and the Stewart Platform as a whole.

Individual actuator position
All six actuators are presented with different commanded trajectories. As can be seen in Figure 2.7, the commands include some complicated and rapid motions. Figure 2.8 shows that the following error of each actuator remains in the order of *1 mm* throughout, for all six actuators. Its peak values (close to *3 mm*) take place at the point where the commanded motion is most demanding, around the *20 s* mark.

These values were obtained with data from the encoders (sampled every *50 ms*), which were converted from pulses to revolutions, and subsequently to linear displacement through the screw lead.

Platform position and orientation
For the purpose of controlling the whole Stewart platform, it is important to evaluate the control system's capacity to produce the desired position and orientation of the top platform relative to the bottom. The following two figures show these results: Figure 2.9 has the linear position commands and errors in the frontal, *X*, lateral, *Y*, and vertical, *Z*, positions; Figure 2.10 contains the respective angular commands and errors about the three axes: roll, *X*, pitch, *Y*, and Yaw, *Z*.

These values were obtained by employing the direct kinematics iterative algorithm. The 'measured' actuator lengths were taken from encoder data, while the initial guess was defined for the first iteration from a well-known initial position of the platform. Consequent iterations – spaced about *50 ms* apart – were started at the position where the previous one had converged, with a convergence radius of 10^{-6} *m*.

Linear acceleration and angular velocity

The reason for using the Stewart Platform in this thesis is, ultimately, to simulated motion. And in doing so, the most important variables are linear accelerations and angular velocities. This is why the two IMUs were mounted on the platform.

In this case, Figure 2.11 shows the commanded specific forces and angular velocities of the platform in the frontal and lateral directions, as well as the estimated values of the Barnacle and SEN inertial units. Figure 2.12 shows that even the direct kinematics position data can be derived to obtain a very good estimation of these variables, consistent with the direct measurements taken by the IMUs.

The vertical axis was not included because whenever one wishes to simulate motion in the *X*- and *Y*-axes, the linear and angular motion of the *Z*-axis becomes mostly a salve of the former. This is due to the fact that the gravity vector is being used to simulate motion; yawing does not alter the simulator's attitude –orientation relative to the earth's gravity- and vertical specific force depends on how much stimulus is needed in the other two directions.

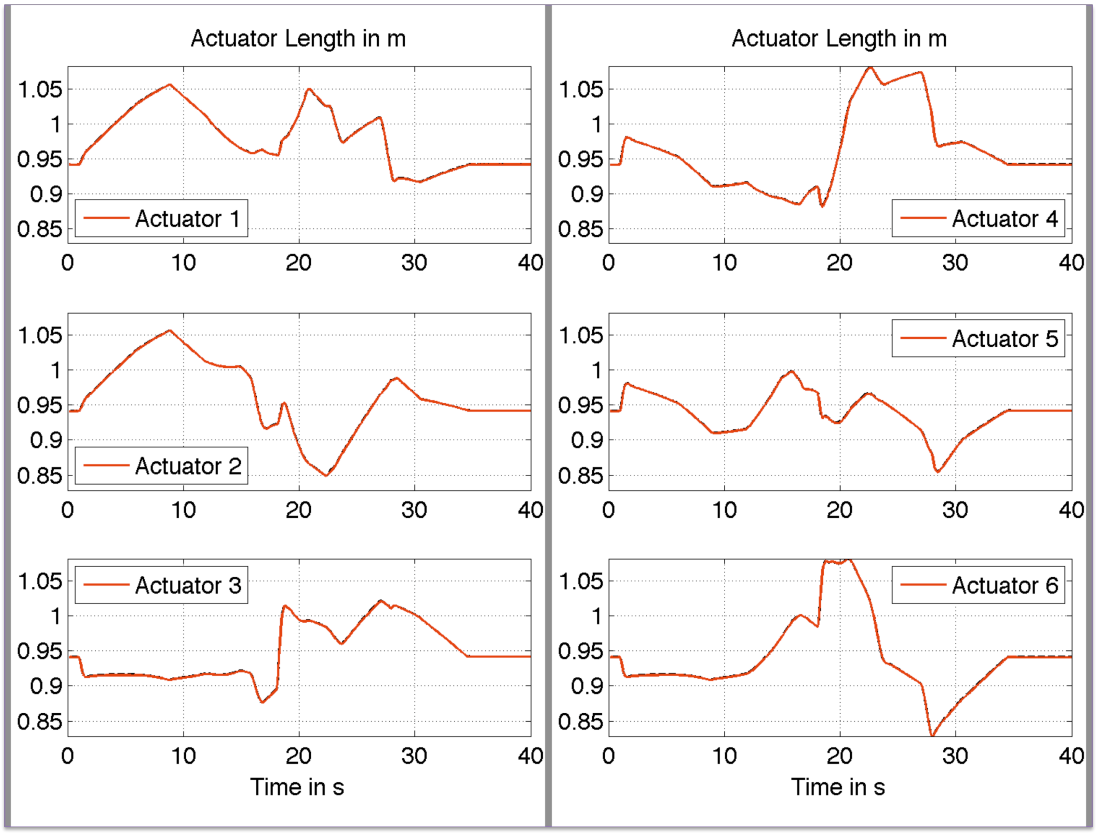


Figure 2.7 – Individual actuator commanded motions

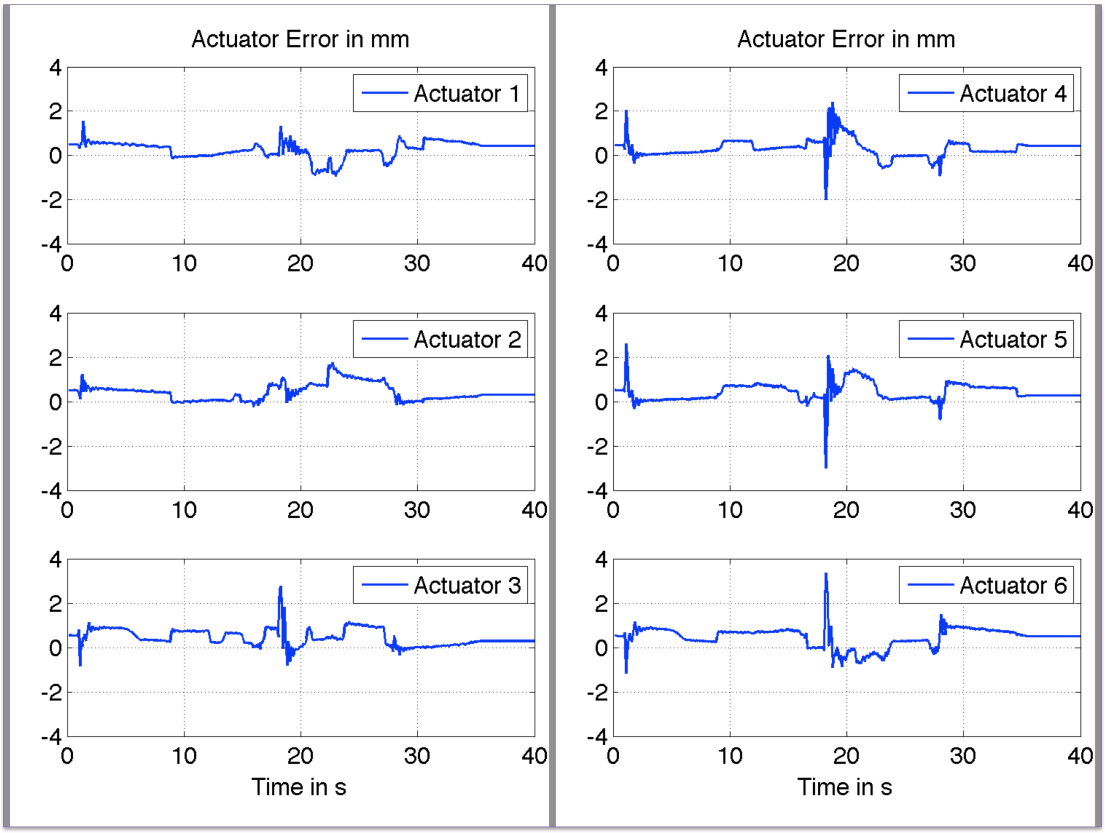


Figure 2.8 – Individual actuator following error

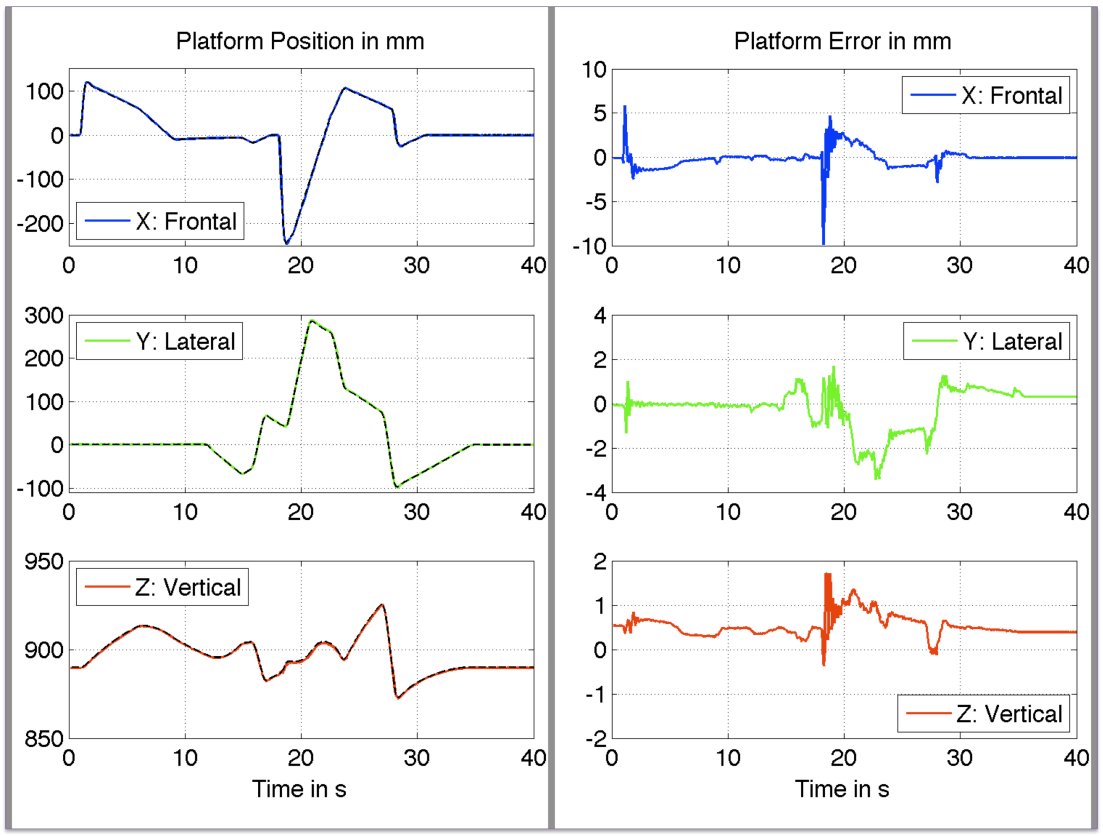


Figure 2.9 – Stewart Platform linear position

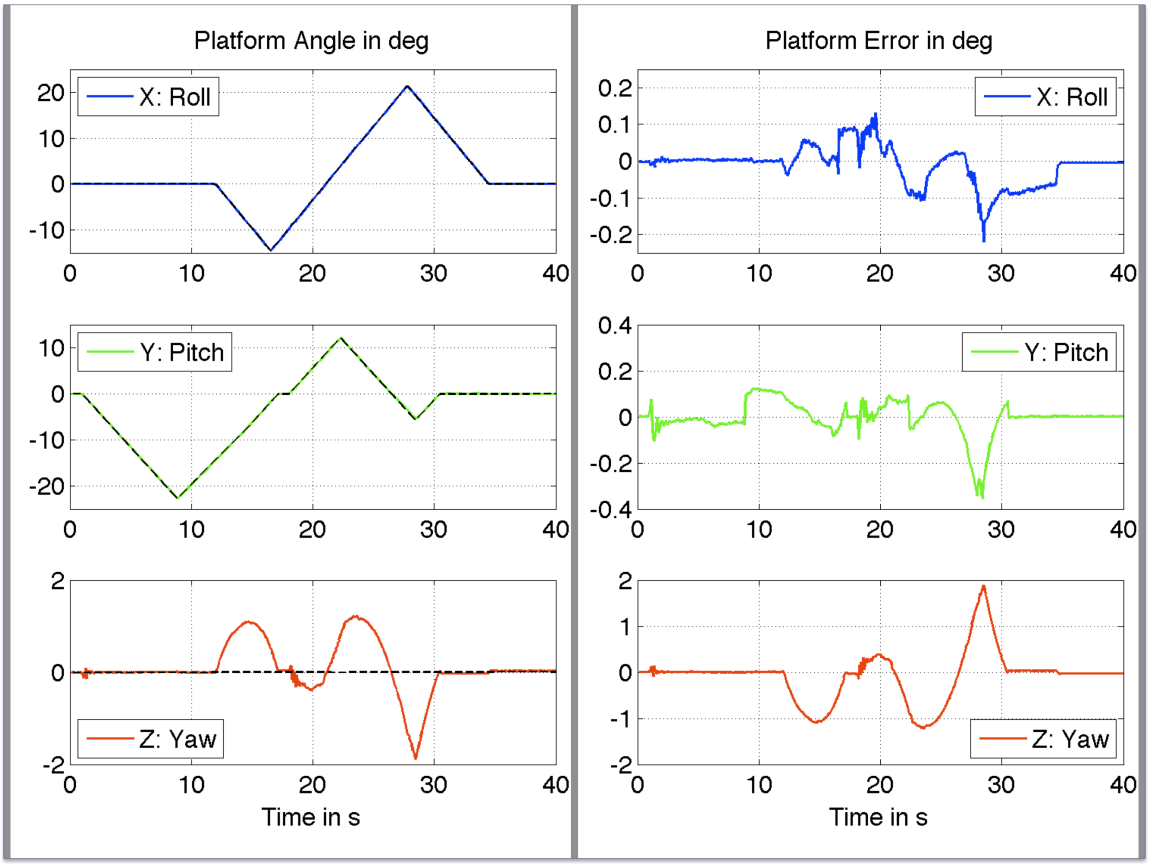


Figure 2.10 – Stewart Platform angular position

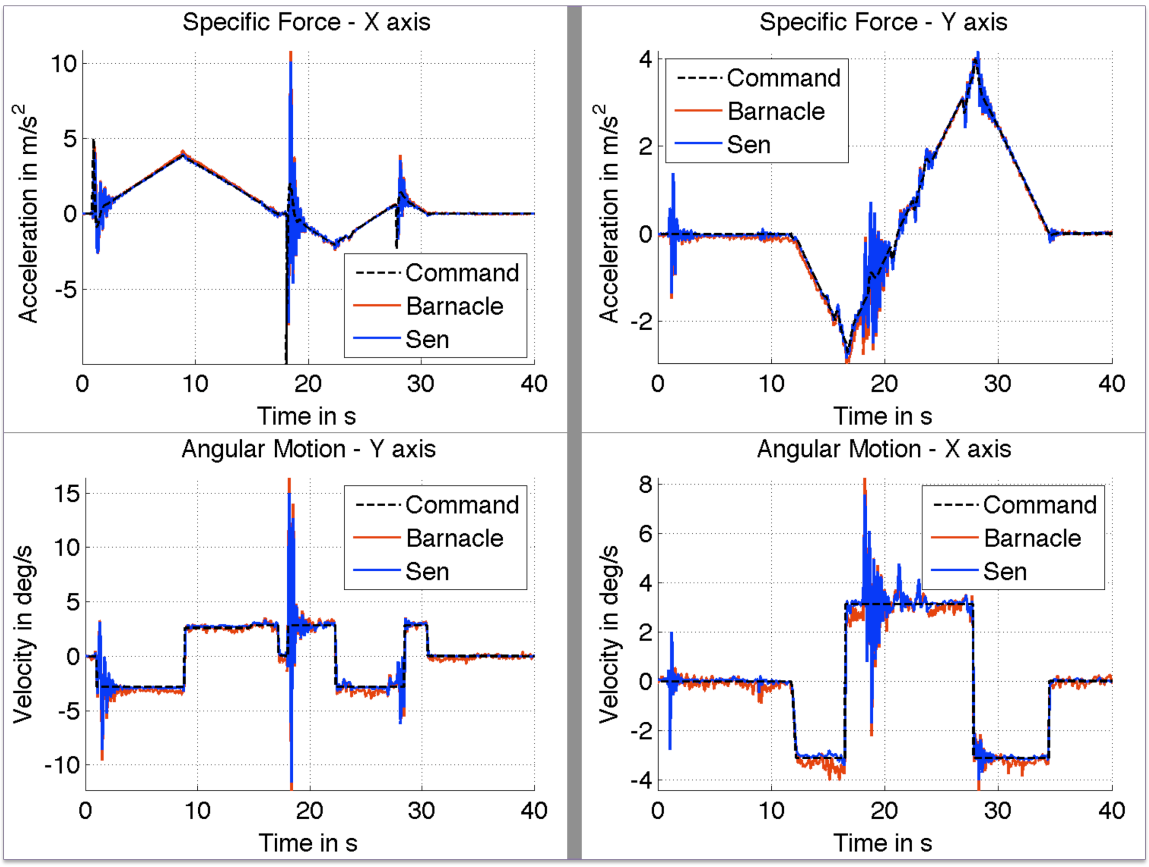


Figure 2.11 – Stewart Platform specific force and angular velocity (IMUs)

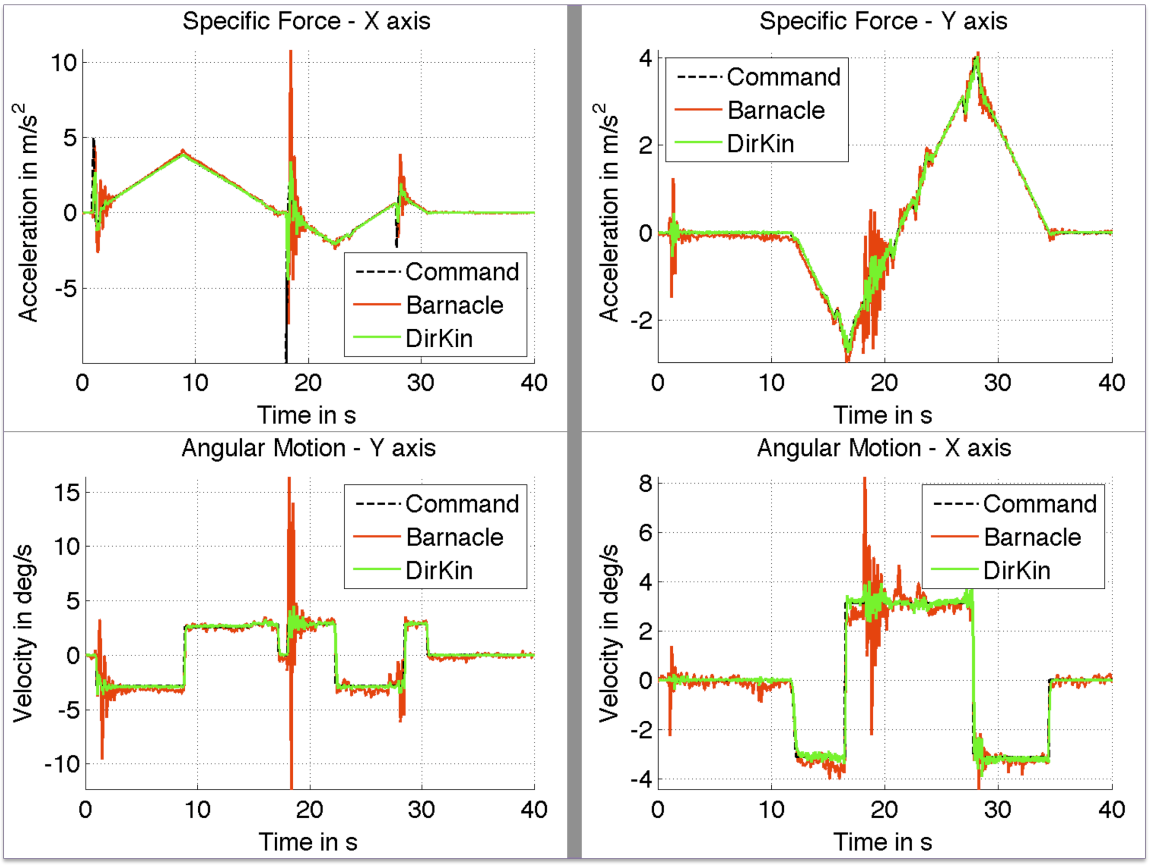


Figure 2.11 – Stewart Platform specific force and angular velocity (Direct Kin)

2.6 Discussion

The robust control approach proved successful to the extent to which it is needed in this work. A more thorough set of trials and theoretical tests could be sought in the future. These would permit a deeper understanding of the capabilities and limits of this method. None the less, this is better than simply tuning the gains through trail and error, as we avoid putting the real system at risk. The robust method provides the engineer with better confidence that the controller will work for the majority of the required working conditions.

These experiments also allowed the characterization of the different measuring systems that were considered to estimate the platform's kinematic behaviour. Sensing the state of this six degree-of-freedom parallel robot proved to be more difficult than meets the eye. Kalman filtering or other sensor fusion techniques could be applied to obtain more reliable estimations, which may enable, for example, the inclusion of a feedback channel for the motion controller based on platform acceleration and angular velocity. This would draw us closer to defining a controller for the Stewart platform, as opposed to a controller for each linear actuator.

3. MOTION CUEING ALGORITHMS

3.1 Introduction and State of the Art

The integration of motion simulation with high quality video and sound has been an active field of development due to its application in training and entertainment. The ability to recreate the dynamics of different vehicles makes motion simulators very attractive devices for training individuals as drivers or pilots. In such cases, whereas sensory information related to the perception of motion is relevant, it can be veiled by audiovisual interaction and by learning how to read instruments and manipulate controls accordingly. This allows for positive transfer of training to take place without a hi-fidelity recreation of the physics involved (Kihl, Herring, Wolf, McVey, & Kovuru, 2006).

A special case of training arises when passengers who have no control over vehicle motion must perform certain on-board tasks. As they don't drive the vehicle, they cannot anticipate its dynamic effects and consequently their prediction of their own motion is poor. In this case, recreating the physics of motion perception is of the upmost importance, in particular when said motion interferes with on-board responsibilities, such as shooting from a moving vehicle. In this thesis, a methodology for the definition of simulator motion cues based on this approach presented. This way, the motion simulator may be used as a coaching platform on which subjects can learn to anticipate and offset their own movement, thus improving their performance on said tasks.

Currently, most training-oriented simulator modules are aimed at pilots or drivers. One example is the training program for Arizona State snowplough drivers, as reported in (Kihl, Herring, Wolf, McVey, & Kovuru, 2006). The dangerous nature of operating heavy and costly machinery in blinding snowstorms prompted the implementation of a simulator with very limited motion recreation. (Tseng & Fong, 2000) designed a land vehicle driver simulator based on a Stewart-Platform and a virtual visual system. For the generation of motion signals they applied the Classical Washout Algorithm as developed at the University of Toronto for a large commercial aircraft pilot training program (Grant & Reid, 1997). The Washout Algorithm uses filters in order to separate high and low frequency components of motion and compute the simulator motion cues accordingly. The process by which the filter parameters are set and their function within the algorithm are thoroughly explained in (Grant & Reid, 1997).

So far, none of these references mark a clear distinction between the perception of motion and the role of the vestibular motion sensors in the inner ear. This has been more carefully studied by Holly and McCollum, who formally defined the perception process and its properties (Holly & McCollum, 1996). They introduced the 'Sensor

Space' as a set containing all the possible responses of all the relevant sensors of self-motion to the motion of the body. In our work, this concept has been adapted to serve as a tool in simulator motion design by defining the Sensor-Sate vector. In an accompanying paper (Holly & McCollum, 1996), they also classified Equivalent Motions as those groups of different motions which produce identical responses of the sensors; this analysis was limited to sustained motions in which acceleration (both inertial and gravitational) remains constant. We have redefined this concept in order to use it in computational models so that it may be applied to non-sustained motions as well. These works were based on the detection of the accelerations of the body, which occurs mainly in the vestibular system.

The behaviour of the vestibular organs has been tested experimentally by Goldberg and Fernandez (Goldberg & Fernández, 1976), (Goldberg & Fernández, 1971) and modelled by (Ormsby & Young, 1977), among others. A comprehensive explanation of the biomechanics and dynamics of the organs can be found in (Rabbit, Damiano, & Grant, 2004). A multi-sensory motion perception model was developed by (Borah, Young, & Curry, 1989) where information from the vestibular, visual, proprioceptive and somatosensory systems was integrated by a Kalman filter.

In the present work, the biomechanical response of the vestibular sensors is proposed as the base for the generation of motion cues for simulators. This response is formally expressed in the Sensor-State vector, thus allowing one to differentiate between the subjective process of motion perception as a whole and the reaction of the vestibular sensors. By focusing on this vital part of motion perception, the Sensor-State imitation strategy is defined and tested in a simple example with a planar motion simulator.

The role of motion cues in vehicle simulation is to provide a sensory experience as similar as possible to that of ridding on the real vehicle. The main challenge in this process is to keep the simulator within the kinematic and dynamic capabilities of the motion mechanism. For this reason, it is inevitable that – at various moments during the simulation – the simulator motion will not provide a good recreation of the motion stimuli produced by the vehicle. This thesis proposes a method to quantify such simulation inaccuracies, in a way that directly relates simulator motion to its impact on perceived motion aboard the device.

There are two main approaches to calculating simulator motion cues. Washout algorithms (Grant & Reid, 1997) use filters to eliminate the parts of motion outside of mechanism capabilities and help return the platform to its central (or neutral) position after displacements. More recently, human centred algorithms, which explicitly use vestibular sensor models to compute simulator motion, have been successfully employed (Telban & Cardullo, 2005). In this approach, a controller computes the linear acceleration and angular velocity of the simulator based on vestibular sensor response error. This error is measured as the difference between sensor response to vehicle and simulator motion.

Since the controller uses this sensor response error, it is vital to quantify and characterize it. In this sense, we propose to use a set of quantitative indicators which directly relate the simulator motion cues to the errors in sensor response and, ultimately, to their effect on overall motion perception aboard the simulator.

The advantages of using these indicators are illustrated both in simulation and experimental tests in which simulator motion cues generated through different algorithms are compared. Finally the possible applications of the indicators are discussed.

3.1.1 Classical Washout Algorithm

This method is the one most commonly used in the motion simulation industry, and thus it merits a closer, if brief, explanation.

The Classical Washout Algorithm has been the benchmark for simulator motion cue generation since the mid 80's (Grant & Reid, 1997). It uses filters to eliminate parts of the real motion that cannot be imitated on a simulator due to kinematic and dynamic limitations. In addition, it returns the platform to its central position after linear accelerated motions lead to its displacement (it 'washes out' the displacement of the simulator, hence the name).

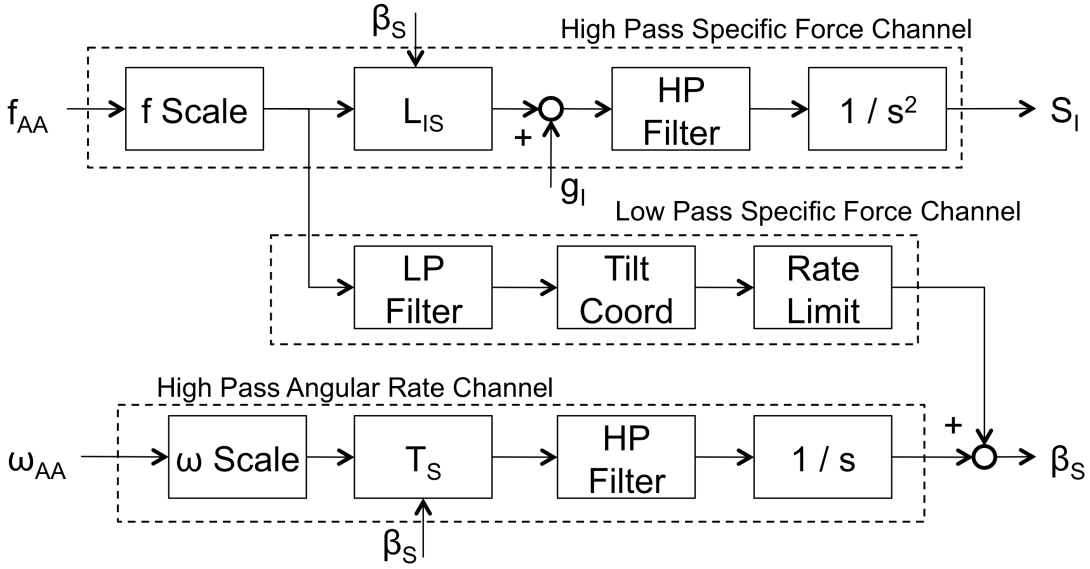


Figure 3.2 – Classical Washout Algorithm

The system's inputs are the vehicle specific force (in m/s^2, acceleration minus gravity) and angular velocity (rad/s). These signals can come from real vehicle data or a dynamic model of its motion. Its outputs are the simulator's linear acceleration (m/s^2) and angular orientation (rad).

Motion cues are provided through three channels, all of which use linear, 2nd order filters. Channel 1 high-pass filters the specific force to produce high-frequency linear accelerations. Channel 2 low-pass filters the specific force; then the tilt-angle is computed as the ratio between the filtered specific force and the gravity magnitude (with small-angle approximation: $\sin\vartheta x = \vartheta x = fy/g$; $\vartheta y = fx/g$); this coordinate is rate-limited so that it occurs at a low angular velocity in order to avoid it being sensed by the passenger's semicircular canals. Finally, channel 3 high-pass filters the angular velocity stimulus to provide high-frequency angular rate cues; these are integrated to produce the corresponding Euler angles. The outputs of channels 2 and 3 are added to provide the total angular orientation signal (low plus high frequency).

The 2nd order washout filters must be tuned in terms of their natural frequencies and damping ratios so that the algorithm produces accelerations and displacements within the mechanism's capabilities. Since there are three channels and each channel has three filters (one for each direction in 3D space) there are 18 parameters to tune.

Additionally, the specific force motion is imitated through both high-frequency linear and low-frequency angular motion. Thus, the low-pass and high-pass cut-off frequencies must be chosen carefully.

For this work, a different approach has been taken to the definition of the motion cueing problem. The washout algorithm was implemented on MATLAB Simulink software © in order to analyse the sensory response it induces in the simulator passenger. This way we can compare its results to those produced by the proposed Sensor-State Controller approach.

3.2 Theory of Motion Perception

According to Holly and McCollum [4], the process of self-motion perception is divided into two successive stages: a physical one and a nervous one. The first, regulated by the laws of physics and the biomechanics of the motion sensors, consists of the transduction of actual motion to a set of stimuli sent from the sensors to the brain. The second relates to the integration of such sensory information in the brain to produce a conscious perception of motion. These two stages will be referred to as Physics Function and Nervous Function, respectively.

As the second stage is inherently subjective, it could be more interesting to use motion simulators for reconstruction of the **sensation** of motion, and not the motion itself. Because the Physics function is many-to-one (Holly & McCollum, 1996), different motions of the head can result in identical signals to the brain, making these motions impossible to distinguish, or **equivalent**. Finding these collapses of the Physics function is vital due to the limited motion space of the simulator, which makes imitating the exact dynamics of an aircraft or motor vehicle impossible.

Extension of the Framework for Motion Perception

In order to incorporate the self-motion perception into a motion simulator, the two previously described functions have been chained with a preceding function, which relates to the simulator motion (Figure 3.3).

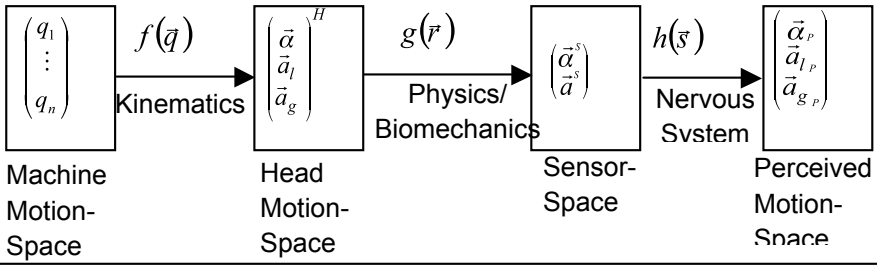


Figure 3.3 – Framework for the motion of an *n* degree-of-freedom simulator mechanism

This first function –called Kinematics- is defined in the machine motion-space. If the machine has *n* degrees-of-freedom, this function maps their motion onto the head's motion. The kinematics function can be expressed in terms of the configuration vector of the motion simulator $(\vec{q})$ as:

$$\vec{r} = \begin{pmatrix} \vec{\alpha} \\ \vec{a}_l \\ \vec{a}_g \end{pmatrix}^H = f(q_1 \ldots q_n) = f(\vec{q}) \qquad (3.1)$$

Sensor-State Vector

Since our primary concern is the mechanism dynamics, a Sensor-State vector will be defined in terms of vestibular variables. Additional information from vision, hearing and the somatosensory system will not be considered. The Sensor-State vector $\vec{s}$ is a 6-dimensional vector containing the responses (vestibular afferent firing rates – VAFR) of the Semicircular Canals (SCC) and the Otoliths (OT) to rotational and linear gravito-inertial accelerations of the head, respectively:

$$\vec{s} = \begin{pmatrix} \vec{\alpha}^S \\ \vec{a}^S \end{pmatrix} = g(\vec{\alpha}^H, \vec{a}_l^H, \vec{a}_g^H) = g(\vec{r}) \qquad (3.2)$$

It should be noted that gravitational and inertial accelerations are impossible to discern in the OT, therefore no distinction between them is available in the Sensor-Space. As previously mentioned, the dynamics of these organs have been widely studied and modelled. Using such models, it is possible to **estimate** the Sensor-State.

We propose to use this estimated Sensor-State in order to **compare the sensation** induced by the simulator motion (simulator Sensor-State) and the one induced by the motion which must be simulated (reference Sensor-State). The former should be as similar as possible to the latter.

In contrast, the process by which sensory information is combined and interpreted is difficult to model and highly variable among subjects (Holly & McCollum, 1996). The Nervous function can be expressed as:

$$\vec{p} = \begin{pmatrix} \vec{\alpha}^P \\ \vec{a}_l^P \\ \vec{a}_g^P \end{pmatrix} = h\left(\vec{\alpha}^S, \vec{a}^S\right) = h\left(\vec{s}\right) \qquad (3.3)$$

Such definition of the framework, allows us to establish a difference between the state of the motion sensors and the perceived motion.

3.2.1 Motion Cueing as an Engineering Problem

As a result, a formal delineation of the simulator's objective may be proposed:

> *The kinematics of the machine should be defined so that the **Sensor-State** of a passenger on the real vehicle is imitated as exactly as possible.*

This way, both the Kinematics and Physics can be calculated to provide a precise sensory experience, while avoiding dealing with the subjective nature of the Nervous function.

This is of particular importance since we are looking to train the subject's ability to interpret sensory information and construct an accurate perception of motion. By continuously exposing him to a certain sequence of motions and their consequent Sensor-States, his Nervous function can be trained to produce a more accurate perception of self-motion and orientation, in order to optimize on-board task performance.

3.3 Proposed Approach

Based on this theoretical analysis of the basic properties of motion perception in the human body and to the resulting statement of the motion cueing problem as an engineering problem, devising a solution in the form of a Sensor-State controller is natural. This section will show its role within the motion cueing system, the proposed structure, and each of the components of such a controller.

3.3.1 Motion Cueing as a Control Problem

A possible solution for the engineering problem stated above can be interpreted as a control problem with the structure shown in Figure 3.4.

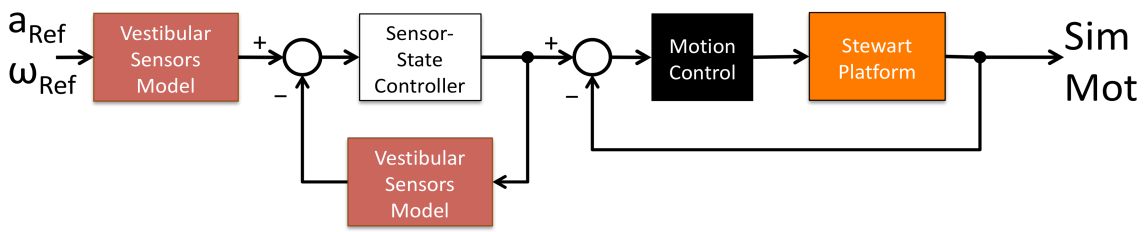


Figure 3.4 – Sensor-State Controller in motion cueing system

In this sense, the controller should be connected in series with the motion system, so as to provide it with the relevant motion commands. Such commands are defined in terms of the difference in the estimated of the Sensor-States aboard the real vehicle and the simulator. Therefore, Vestibular sensor models must be carefully chosen. Consequently, the inner structure of he controller can be defined.

3.3.2 Vestibular Sensor Model

First of all, a vestibular sensor model must be chosen in order to implement the proposed Sensor-State feedback system.

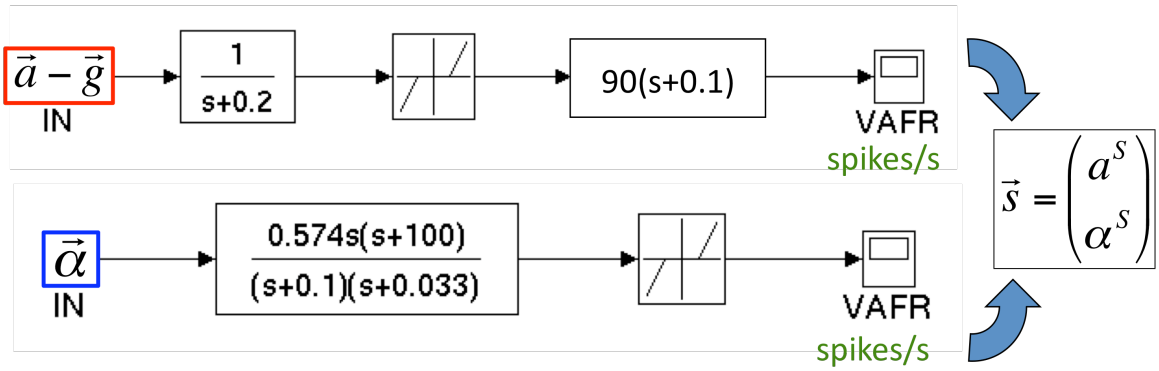


Figure 3.5 – Vestibular sensors model from (Borah, Young, & Curry, 1989)

The vestibular sensors have been widely studied and modelled in the past four decades. In the model employed in this paper (Borah, Young, & Curry, 1989), each organ is modelled independently as shown in Figure 3.5; their input is specific force (linear acceleration minus gravity) for the otolith and angular acceleration for the semicircular canals. Their output is the Vestibular Afferent Firing Rate (in spikes per second) of each organ in each direction. By grouping the VAFR of both organs in one (six-dimensional) vector we obtain the Sensor-State vector, which completely identifies the response of the vestibular sensors to the motion they are subjected to.

An important element of the model is the threshold effect, which shows that small stimuli do not produce a reaction of the sensor. It should also be noted that the otoliths don't distinguish between motion-induced and gravity-induced stimuli.

This vestibular sensor dynamic model will be used in order to analyze and compare sensory response to vehicle and simulator motion.

This model was chosen, among many others available, for two main reasons. First of all, while the order of transfer functions is based on theoretical analysis of the organ's behaviour, its parameters have been tuned to represent the vestibular dynamics reported in the experiments of Goldberg and Fernandez in the 1970's. Therefore it combines both theoretical and experimental validation.

Second, it includes – unlike many others: (Rabbit, Damiano, & Grant, 2004), (Ormsby & Young, 1977) – not only the deformation of the organ due to accelerated motion, but also the transduction of this deformation into an electrical signal that travels through the nervous system to the brain. In this sense, it fits better into the framework of motion perception we have defined for motion simulators. Therefore, by combining the responses of the otolith and canal models, a sensible estimation of the Sensor-State vector can be provided.

3.3.3 Sensor-State Controller

Thus, the Sensor-State Controller - based on the human centred approach – can now be introduced. It uses the vestibular response error – vehicle Sensor-State minus simulator Sensor-State – in order to calculate simulator linear acceleration and angular velocity. The controller is designed to have simulator vestibular response track the real vehicle response, while taking into account the kinematic and dynamic restrictions, as well as other nonlinear effects like angular velocity saturation in tilting motions.

Like the washout algorithm, it has three channels, shown in Fig. 3.6. Channel 1 high-pass filters the otolith response error and uses a proportional controller to produce simulator linear acceleration. Channel 2 low-pass filters otolith response error and uses a proportional-derivative controller to produce low-frequency angular velocity of the platform. This angular velocity is saturated so that it falls within the sensor threshold. Channel 3 high-pass filters semicircular canal response error and uses a proportional controller to produce simulator high-frequency angular velocity.

The most important difference with the washout is that, in this case, simulator motion is produced through closed-loop control of the vestibular response. In this sense, the trajectories computed by the controller take into account the vestibular organ dynamics. The calculations also include the effect of producing rotational simulator

motion that is not centred in the head – which may induce unwanted linear (centripetal and transverse) accelerations of the head.

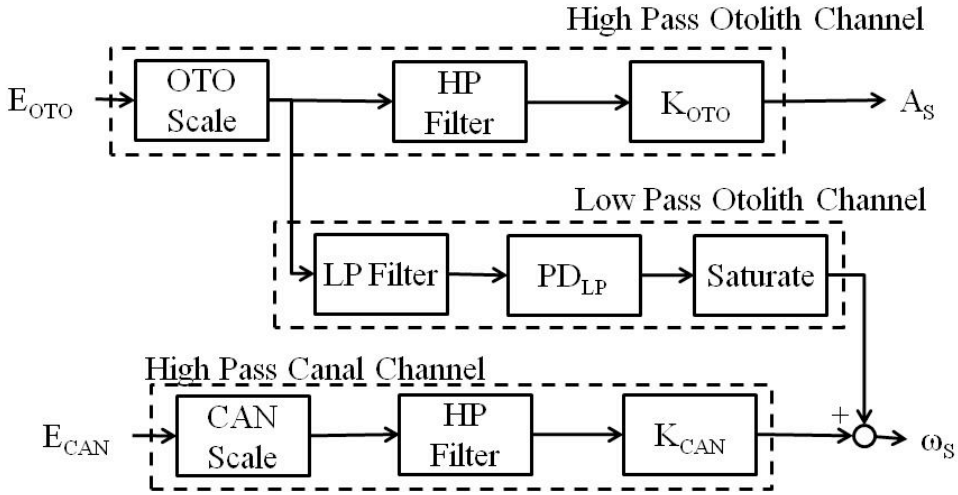


Figure 3.6 – Sensor-State Controller

The proportional gains in the high-frquency channel are mainly used to convert from the sensor-state signal (in spike/s) to acceleration commands. However, they can also be used as a scaling factor, or to choose how much prominence the high-frequency content should have.

The proportional derivative controller was used in the low-frequency channel because of the saturation element. Due to the fact that low-frequency tilts are not supposed to be detected by the vestibular system, the saturation limit on the angular velocity is rather low. A PD controller – and the derivative gain in particular - enables the algorithm to modulate the speed at which the tilts are executed depending on the nature of the otolith sensor-state error, instead of just going at maximum velocity all the time. Such a motion would produce unwanted jerkiness aboard the simulator.

3.4 Motion Cue Evaluation

In order to better understand the relationship between differences in the vehicle and simulator motions and their impact on the quality of the simulation, quantitative error indicators were defined.

The starting point was a qualitative description of the common types of motion cue errors in terms of specific force presented by (Grant & Reid, 1997). This definition was translated to quantitative indicators based on vestibular response errors. The two comprehensive indicators that were created provide a helpful tool in discerning amongst good and bad motion simulations.

This section presents these indicators and explains how they are applied and interpreted.

3.4.1 Error Classification

(Grant & Reid, 1997) defined three classes of motion cue errors in simulators based on comparing the simulator and vehicle accelerations, and discussed their relative impact on perceived motion simulation. Figure 3.7 shows examples of the three types of cue errors described below.

“False cues” occur when simulator acceleration has an opposite sign to the vehicle acceleration at any given moment. For example, while simulating a positive frontal acceleration, the simulator must decelerate to avoid reaching its kinematic limit. Another case of a false cue is when a sustained motion presents a relatively high-frequency distortion in the simulator. This type of error is the most harmful to motion simulation fidelity. These two examples can be seen in Figure 3.7.

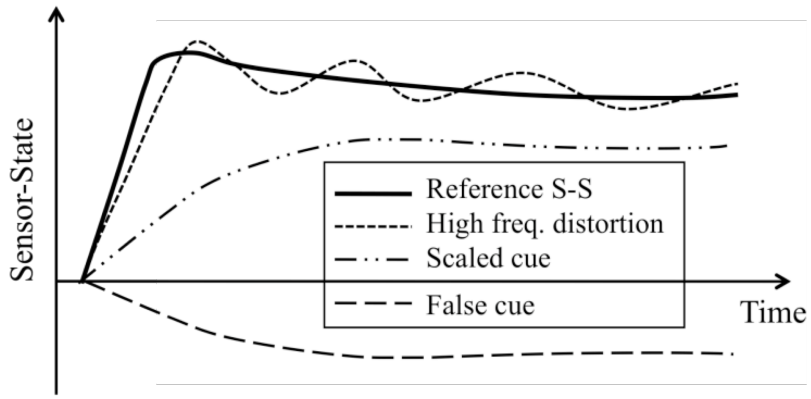


Figure 3.7 – Motion Cue Errors

“Scaled cues” occur when both the simulator and vehicle accelerations are in the same direction, but with different magnitudes – usually the former being smaller. While they still indicate that the motion cue differs from the ideal scenario, its effect on perceived motion fidelity is not as drastic.

Finally, “phase errors” cause the least problems to motion perception fidelity and occur when the vehicle and simulator motions are out of phase.

We use these qualitative assessments as the basis for the definition of objective, quantitative indicators of cue quality.

3.4.2 Basis for Quantitative Sensor-Based Evaluation

After reviewing the aforementioned qualitative classification, it is clear that common error indicators cannot be directly applied to the overall motion cue in order to judge

on its quality. If we were to use, for example, the 2-norm of the error signal, the result would not be a good indicator of how well the motion cue is simulating the vehicle motion. This value does not take into account the inherent properties of the process we wish to control, which, ultimately, is motion perception aboard the simulator.

In this sense, the calculations involved in determining quantitative performance indicators must contemplate the error classification described above. Also relevant to this application is the fact that simulator motion is calculated as a function of Sensor-State error. Therefore, understanding what this error means in terms of simulator motion cue quality is crucial.

Taking all of this into account, we propose that quality indicators be based on the Sensor-State error – as opposed to the specific force and angular acceleration errors – and that the quantitative evaluations of this error should be calculated separately for false and scaled cues, in order to gain better insight on how motion cue mistakes are impacting perception fidelity. Also, phase errors could be measured, or their effect could be reflected in terms of false and scaled cues. This way, the quantitative indicators serve as a link between simulator motion, vestibular sensor response and the overall quality of the perceived motion aboard the simulator.

3.4.3 Three Basic Elements

In quantifying the error we are interested in three elements of the signal: time, magnitude and variation. In other words, how much time did the simulator spend giving faulted cues, how large were these errors and how much did the errors change over time. This last element is significant because one would rather have a cue that is scaled but maintains a constant error, than one that is not scaled but has strong fluctuations around the reference signal, like the high frequency distortion shown in Figure 3.7.

Three basic measurements can be taken to quantify these elements. For time, one could detect when each type of error is occurring and measure the total time for false and scaled cues separately. For the magnitude, the maximum absolute error in a certain time window could be used. Meanwhile, variation could be denoted by the maximum absolute value of the error derivative, within a certain time window. The respective values would be computed separately for parts of the signal presenting false and scaled cue errors.

Nonetheless, these quantities provide no information about how the error (or its derivative) behaved while it lasted. We may have its time of duration and maximum values, but we don’t know what happened when it was below the maximum.

3.4.4 Two Comprehensive Indicators

To obtain a more complete picture of how the motion cue error is affecting motion perception over the whole time domain, the use of comprehensive indicators, which combine information for time and the other elements, is suggested.

These comprehensive indicators, along with the time duration data of each type of error, can give the simulator designer a very good idea of how the motion cues are performing. And, most importantly, the information is obtained without resorting to subjective input from a pilot ridding the simulator or graphical analysis of the cue.

The Magnitude Indicator. This combines information from time and the size of the error, as shown in Eq. (3.5).

$$MAG = \int_{t_0}^{t_f} \left| E(t) \right| \cdot dt \qquad (3.5)$$

The indicator is, in effect, a weighted sum of the error's magnitude throughout the time it lasted. In this regard, the computed value contains information of both how long the false or scaled cue lasted, and how the error behaved during this time span. This allows one to note the difference between two error signals with identical time durations and maximum values, but different behaviours below the maximum. Even further, when used with the time duration data, one can tell the difference between a long lasting error of small magnitude and a short lasting error with a large magnitude.

The t_0 and t_f values in Eq. (3.5) can be adjusted so that the *MAG* is calculated for time intervals in which only one type of error (false or scaled) is present, therefore isolating the different error types defined in the previous section. Figure 3.8 shows how a motion cue used to simulate a 1 m/s^2 step in linear acceleration can be separated in different time sections and illustrates the advantage of using the *MAG* as a quantitative indicator of the error's magnitude and duration.

As seen in Figure 3.8, the motion can be separated in five sections. First, the initial milliseconds show a good Sensor-State imitation through a linear accelerated motion of the simulator. After this and until 0.19 s there is a scaled cue as the Sensor-State continues to be positive, but is decreasing in magnitude. Eventually the otolith response becomes negative between 0.19 s and 0.45 s as the simulator decelerates to come to a stop and return to the centre of its workspace, thus providing a false cue. As this motion is taking place, the simulator is performing a tilting motion, whereby the gravitational stimulus plays the role of linear acceleration. From 0.45 s to 1.97 s the sensor response to the gravitational stimulus continuously increases, resulting in a scaled cue. Finally, the tilting motion is completed and the vehicle Sensor-State is, once again, correctly imitated with only a small, scaled cue error from 1.97 s onwards.

The *MAG* indicator for false cues would be area A_3, while the value for scaled cues would be the sum of areas A_2, A_4, A_5.

This indicator can also identify and quantify the negative effects of an out-of-phase periodic motion cue cantered on zero – the third type of error defined previously. For small phase differences, most of the motion cue error is assigned as a scaled cue, while only a small part classifies as false.

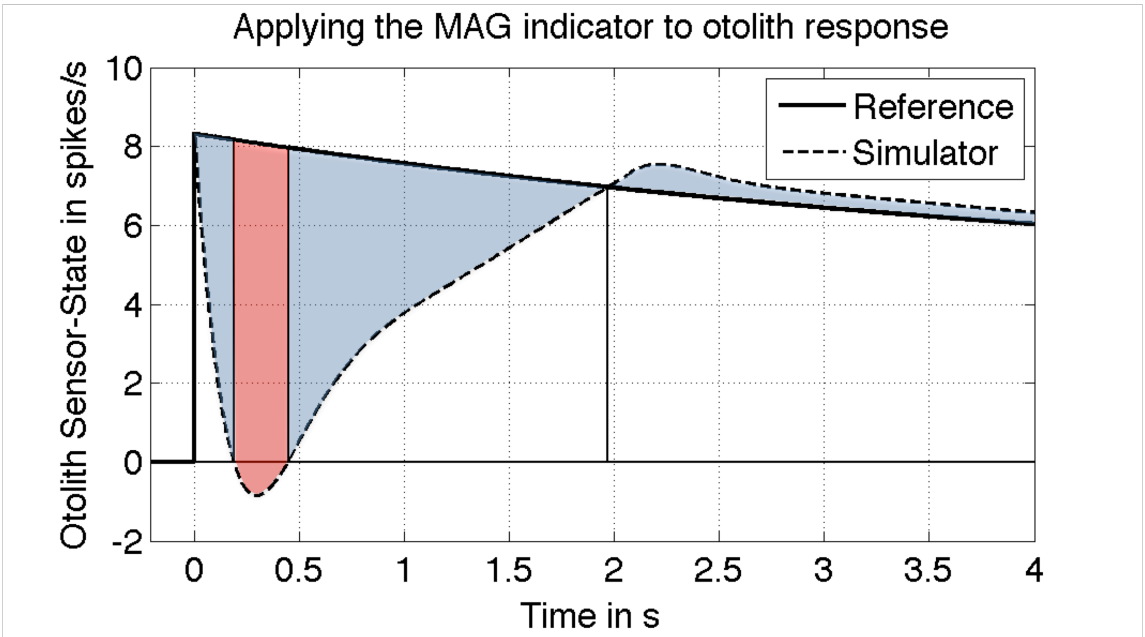


Figure 3.8 – Applying the MAG indicator to the simulation of a linear acceleration step of 1 m/s^2

As the phase error increases, a larger portion of the signal is identified as false, while the scaled part decreases. Because the false cue is considered worst than the scaled cue, the indicator reflects the fact that a larger phase error produces a larger problem with perception fidelity aboard the simulator.

The Variation Indicator. It is computed from Eq. (3.6) and it combines information from time and the error variation.

$$VAR = \int_{t_0}^{t_f} \left| \frac{dE(t)}{dt} \right| \cdot dt \qquad (3.6)$$

The error derivative's absolute value is a measure of the magnitude of the error variations. By integrating such magnitude, the *VAR* indicator performs a weighted sum that expresses how much the error varied in a certain time interval. Again, the values of t_0 and t_f in Eq. (3.6) are selected so that the interval includes only one type of motion cue error.

In this fashion, the *VAR* is revealing how similar the reference Sensor-State and its simulator counterpart are, regardless of the differences in their magnitude. Figure 3.9 shows how a positive reference Sensor-State is simulated by two different signals. The first, of almost identical magnitude, fluctuates significantly around the reference, while the second has a much smaller magnitude, but the same shape.

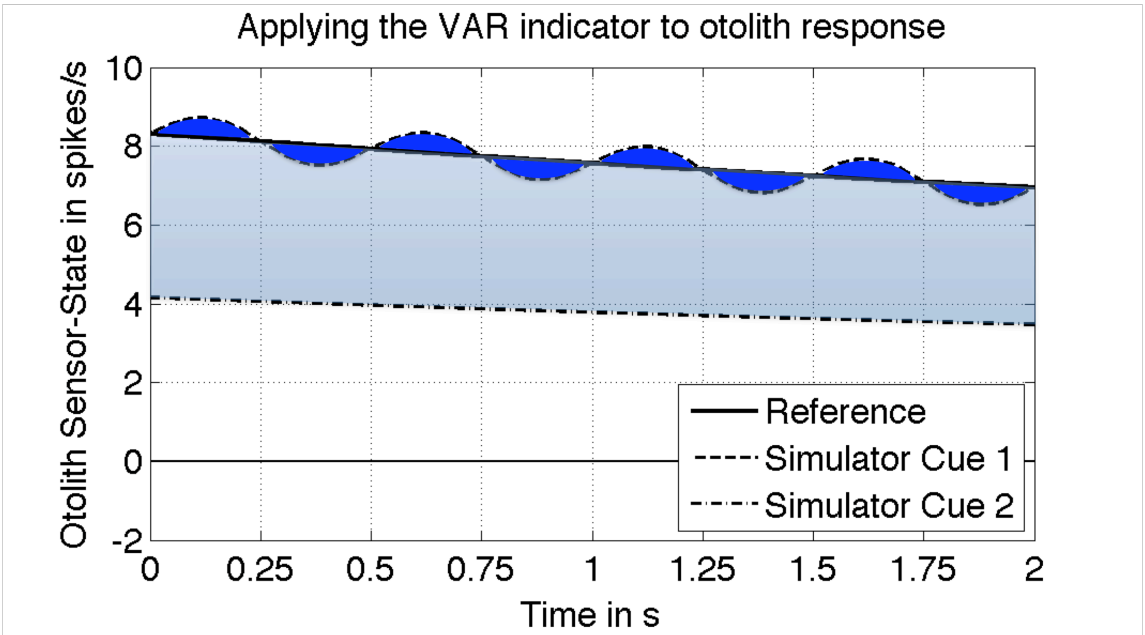


Figure 3.9 – Applying the VAR indicator to the simulation of a sustained motion cue

When these different motion cues, which classify as scaled errors, are compared using only the *MAG* indicator, signal 1 in Figure 3.9 would obtain a lower (therefore better) value. However, this does not account for the false cues associated with high frequency distortions of sustained signals.

Therein lies the relevance of the variation indicator. This type of false cues could be effectively identified from the higher value of the *VAR* for signal 1, as compared to signal 2. Because the reference and this latter Sensor-State have a similar shape, the error remains almost constant hence its derivative is practically null and so is the *VAR* value.

This indicator can also account for phased cue errors. If the phases of the reference and simulator Sensor-States are fairly similar, the error will remain mostly constant and the *VAR* will be small. However, as the phase difference augments, so will the error variation, and this will be reflected in the indicator. This is true notwithstanding the difference in signal magnitudes.

One final note on the use of the error indicators is that they are only applied when the Sensor-State error is sufficiently large (i.e., when the error is larger than a previously

defined acceptable margin of error). This margin can be calculated in terms of the vestibular sensor model by applying the method defined in (Rodriguez & Ochoa, 2008). For the current model, the values are ± 0.9 spikes/s for the otolith and ± 4.001 spikes/s for the semicircular canals. For this reason, one could argue that area A_5 in Figure 3.8 (from 1.97 s onwards) should not be included in the calculation of the *MAG* indicator.

3.5 Results

This section shows the most important results obtained for this thesis, as they combine all the components of the motion cueing system studied so far: Stewart Platform, motion control system, motion drive algorithms and motion cue evaluation. Plus, they include work both in simulation and experiments on the real system.

Simulation played a major role in the development of the Sensor-State Controller and the robust controller for the actuators. It was also instrumental in gaining better understanding of the classical washout algorithm and the vestibular sensor dynamics. By programming of dynamic models for the motion controllers, the Stewart mechanism and the different motion cueing algorithms, a powerful simulation tool has been created for future work on the hexapod at COLIVRI Laboratory.

This tool facilitated the testing of the different subsystems independently, as well as their performance when all are combined into a single motion cueing system. The results of such trials can be seen in the following subsection.

All of this, however, would not be useful if its results cannot be replicated in reality. In order to realize fully the objective of this thesis, different cueing algorithms and controller gains were implemented on the parallel platform at the aforementioned facilities. These results are presented in the subsequent subsection.

3.5.1 Simulation

The flexibility of the simulation environment created in Matlab and Simulink allowed the generation of different testing conditions for the system's behaviour. Three of these scenarios are presented.

Comparing Motion Cueing Algorithms
First of all, the motion cueing algorithms were compared in their ability to generate motion commands that produce good simulations of the reference vehicle's motion. This analysis was done without including the dynamics of the controlled motion system and was aided by the use of the quantitative error indicators defined above.

In order to test and evaluate a motion drive algorithm, one can develop a test motion that represents the key aspects of the dynamics one wishes to imitate on the simulator. The algorithm that presents the best results would be better suited to the specific application. For this example, the simulation will involve a racing car on a racetrack. The main elements to be simulated are straight-line acceleration, short curves that require no braking, and complex curves that require both braking at the entrance and acceleration at the exit.

For this purpose, two test motions were designed to include these features, as shown in Figure 3.10. On the top plot, a car starts from rest and accelerates at 5 m/s^2 in the frontal direction (X-axis) for 5 s. As a curve approaches, frontal acceleration decreases until it reaches 1 m/s^2 at 14 s and finally a null value at 15 s. Meanwhile, the car enters a left turn, as shown by the ensuing lateral acceleration (Y-axis) at 11 s. It increases to 4 m/s^2 at 14 s, where it remains for 1 s and finally decreases at a constant rate to zero at 16 s. This is Test Motion 1.

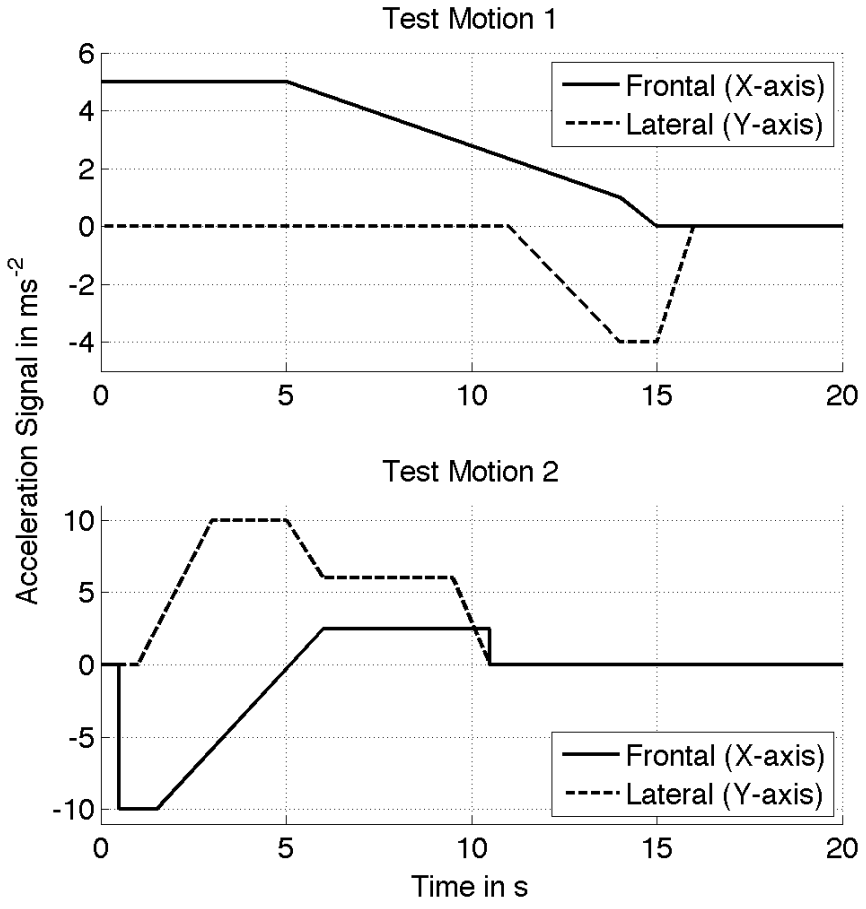


Figure 3.10 – Test motions 1 and 2: acceleration signal

Test Motion 2 is shown on the bottom plot in Figure 3.10. Here, the car breaks heavily before a curve, between 0.5 s and 1.5 s. Then, deceleration decreases from -10 m/s^2

to null between 2.5 s and 5 s. Meanwhile, the strong right turn causes a lateral acceleration that rises to 10 m/s^2 at 3 s. As the car exits the apex, close to the 5 s mark, lateral acceleration falls to 6 m/s^2, as frontal acceleration rises to 2.5 m/s^2 at 6 s. Then, lateral acceleration drops to zero between 9.5 s and 10.5 s, at which point the car exits the curve at a constant speed.

Since the Test Motion described above is for the car's center of gravity, a rigid body kinematics function was used to calculate the passenger's head motion. This head motion is then used as the input to the simulator motion drive algorithms, f_{REF} and ω_{REF} in Figures 3.2 and 3.4. In this case, the simulator motion mechanism transfer function was assumed as unity, followed by another rigid body kinematics function that computes the simulator passenger's head motion. In this fashion, the model accounts for the fact that the passenger has different positions in the car and in the simulator, with respect to the point about which the respective motions are specified.

The passenger's head motion is then used to calculate the Sensor-States, both for vehicle and simulator motion. With both states, one may compare vehicle and simulator sensor response and apply the error indicators to objectively quantify the quality of the motion cues.

Test Motion 1. The simulator motion cues generated by both motion drive algorithms produced the otolith organ responses shown in dashed lines in Figure 3.11. These are plotted against otolith response to vehicle motion (in solid lines).

The simulator initially performs a short accelerated motion in the X-axis (frontal direction) to imitate the high-frequency component at the onset of linear acceleration in the car. This produces the short peak in otolith response seen on the graph, similar for both algorithms. The mechanism then decelerates to come to a stop, causing a sharp valley in the Sensor-State. The false cue associated with the braking motion is smaller for the Sensor-State Controller, compared to the Washout.

Thereafter, otolith response in the X-axis increases steadily as the simulator is tilted at the maximum angular velocity in both cases. However, the Washout overshoots its target, peaking at 8.12 s about 6 spikes/s above the reference signal. On the other hand, the SS-Controller has no such problem. These errors remain almost constant for both until the car's frontal acceleration reaches 1 m/s^2, at 14 s. At this point the reference Sensor-State drops more sharply to 9.76 spikes/s at 15 s, where the car's X-axis acceleration has reached zero, a behavior that is not well emulated by either algorithm. As the sensory signal moves towards its steady-state value of 18.79 spikes/s, the SS-Controller is the first to provide a good Sensor-State imitation – at 16.07 s, versus 16.68 s for the Washout.

Lateral acceleration (Y-axis) response is simulated in similar fashion, but with a smaller presence of high-frequency motions. Both algorithms provide almost identical

cues until 15.2 s. At this point the SS-Controller's cue takes a quicker turn towards the rising reference Sensor-State, reaching it about 0.4 s earlier and with smaller overshoot than the Washout.

Vertical direction (Z-axis) sensor response shows practically the same characteristics as the X-axis. This is explained by the fact that vertical otolith stimulation is reduced during tilting motions. Since the gravity vector is used to introduce stimuli in the X and Y-axes, the simulator cannot faithfully recreate Z-axis response aboard the vehicle. It must be noted, though, that the SS-Controller shows smaller peaks, valleys and overshot, than the Washout Algorithm.

Semicircular canal responses for the vehicle are constant at zero, while both simulator cues produce responses below the acceptable margin of error, and are therefore not plotted.

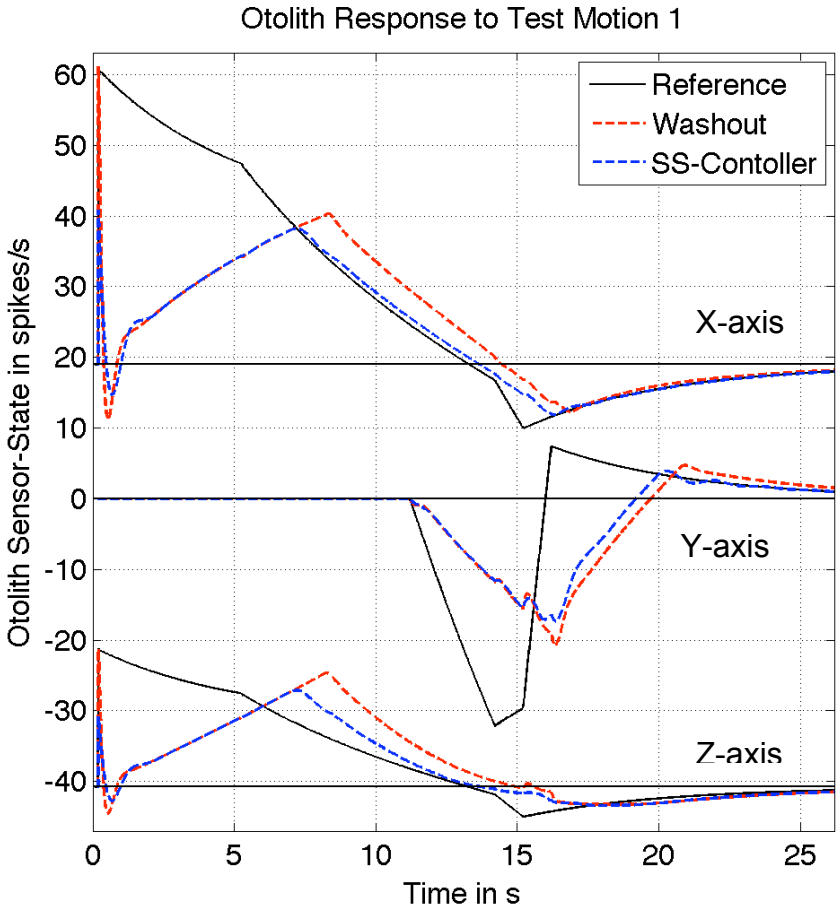


Figure 3.11 – Otolith response to test motion 1

Test Motion 2. Figure 3.12 compares otolith response to Test Motion 2.

The simulation starts with a backward acceleration in the X-axis that emulates the vehicle breaking before the curve. A forward acceleration is then used to bring the machine to a halt. In similar fashion to Test Motion 1, the initial valley in otolith response is similar for both algorithms, but the SS-Controller produces a smaller error while stopping the simulator motion.

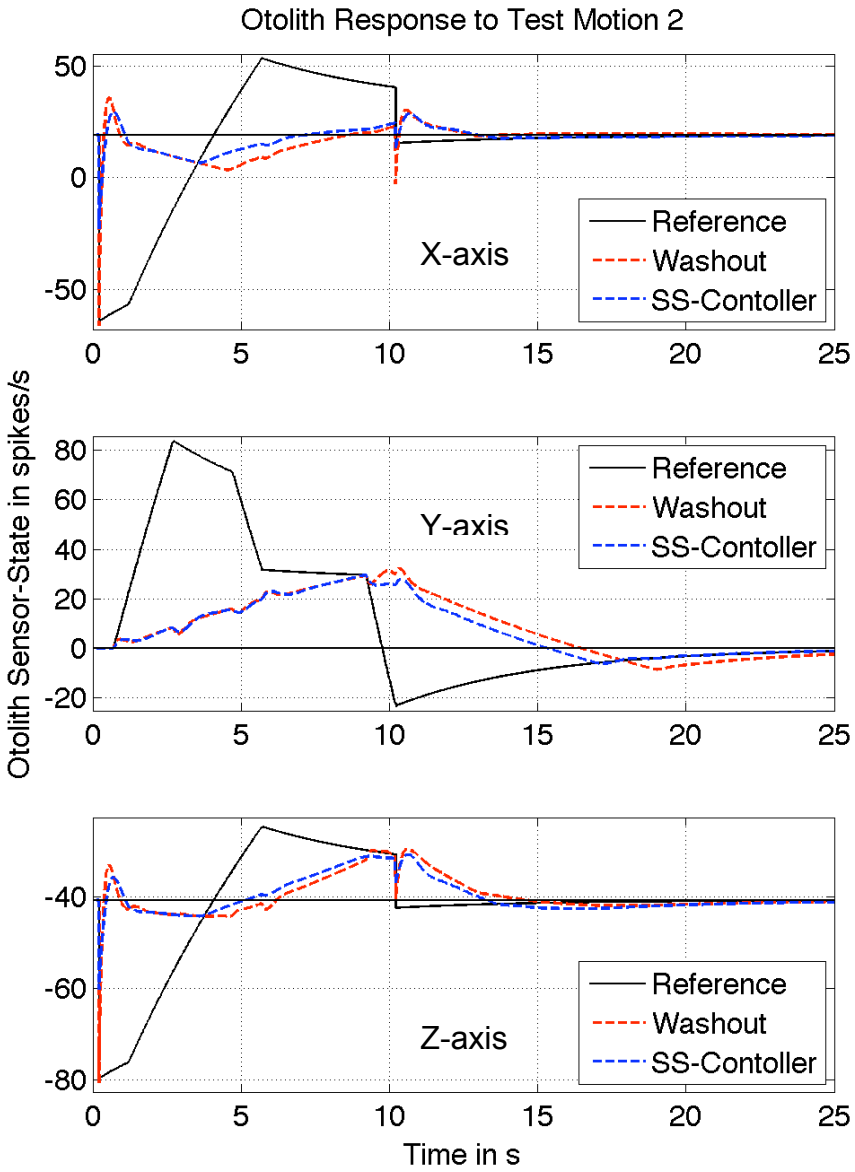


Figure 3.12 – Otolith response to test motion 1

Once again, the tilting motions are identical until 3.55 s, when the SS-Controller takes a quicker turn towards the rising reference Sensor-State. This produces a smaller error than the Washout until 10.5 s. At this point, the sensors react to the sudden suppression of forward acceleration (see Figure 3.11). The Washout overshoots this reaction by almost 18 spikes/s (versus only 3 spikes/s for the SS-Controller) and produces a stronger braking acceleration to stop the simulator after the motion. From 11 s onwards both cues produce similar sensor responses.

Lateral acceleration simulation presents similar results to Test Motion 1: both cues are practically identical, until 9.5 s, when the SS-Controller reacts quicker to the fast fall in reference Sensor-State, thus producing smaller errors. The Z-axis response behaves similar to the X-axis response.

Semicircular canal response is similar to that for Test Motion 1 and is therefore not plotted.

Comparing Motion Cue Quality. As helpful as this qualitative comparison has been, it is subjective and must be performed through a detailed – and time consuming – graphical analysis of the results. Its qualitative nature also means that one cannot reach a definite, objective conclusion with regard to the overall quality of the different motion cues. One can only obtain a general notion of which cue seems to be better in each particular maneuver.

This section will show how, by applying the quantitative indicators, the simulator designer gets an instant, objective vision of how the motion cue performed throughout the simulation. Further yet, since the indicators are directly linked to different motion perception phenomena, they lead to clear-cut conclusions of which motion cue presents the highest fidelity to the reference motion.

Test Motion	Axis	Difference in MAG (%)		Difference in VAR (%)	
		False	Scaled	False	Scaled
1	X	9.95	18.94	45.92	38.76
	Y	19.49	1.73	56.04	3.37
	Z	21.88	25.63	52.95	26.27
2	X	28.88	-19.07	38.68	33.21
	Y	19.05	3.74	31.12	14.40
	Z	29.30	-8.70	44.55	33.86
Combined Sequence	X	23.05	3.51	44.56	38.67
	Y	24.64	-7.86	41.63	16.79
	Z	27.24	7.69	45.20	31.38

Table 3.1 – Algorithm comparison with quantitative indicators

The indicators were computed for each algorithm in three scenarios: each test motion separately plus a sequence in which motion 2 was performed 1 s after motion 1 was completed (at 17 s on the top graph in Figure 3.11). The results for the SS-Controller were subtracted from those of the Washout; this difference is expressed as a percentage of the Washout indicators in Table 3.1. A positive value in the table means

that the Sensor-State Controller had smaller indicators than the Washout, representing superior performance.

For false cues, both MAG and VAR where improved by the Sensor-State Controller for all axes in every test. While the improvements in MAG oscillate mostly between 20% and 30%, those for the VAR – which range from 30% to 55% – are even more impressive. Given than false cues are the most damaging to motion simulation fidelity, this is clear evidence of the SS-Controller's advantage over the Washout.

For scaled cues, the indicators show improvement on all but three cases: MAG indicators for test motion 2 in the X- and Z-axes, plus the combined motion in the Y-axis. In these situations, there has been a trade-off between false and scales cue errors. Since the SS-Controller spends less time producing false cues, it does more time on scaled cues.

Non-the-less, given that the latter are preferable to the former, this is actually a desirable compromise, as long as the reduction in false cues is large enough to justify an increase in scaled cues. This also explains why the improvements are only slight (below 4%) in four other cases: motion 1, Y-axis MAG and VAR; motion 2 Y-axis MAG and combined motion X-axis MAG.

It must be noted that all cases of the VAR have been improved. This can be interpreted as meaning that the SS-Controller's cues have more similar shapes – regardless of the differences in magnitude – than the Washout cues. In this sense, the Controller manages to reduce the presence of high-frequency distortions associated with false cues.

A final important point, is that the improvements where maintained even when the test motions were combined into a single motion. This suggests that one could put bits and pieces of the test motions together in different forms and still obtain better results with the SS-Controller. Since the racetrack can be seen as a combination of straight-line accelerations, simple curves and complex curves, the conclusions from these tests can be expanded for a simulation of the whole racetrack.

Comparing Motion Cueing Systems – CWA vs. SSC

Consequently, once the motion control system was proven to be adequate (as shown in the previous chapter), the cueing algorithms and motion system models were integrated. This way the platform dynamics have been included in the model, instead of assuming it has a unitary transfer function, as before. The second set of trials compared the performance of the Classical Washout Algorithm with that of the Sensor-State Controller under these conditions.

Given the results of the previous case, only one test motion was used for consequent trails. The test motion is simply the combination of the two previously described. For

the sake of clarity, the motion is shown in Figure 3.13. This is exactly the same as the one labelled ‘Combined Sequence’ in Table 3.1.

The otolith sensor-states are plotted in Figure 3.14. The error indicators for this comparison are shown in Table 3.2, after the set of graphs for the next comparison.

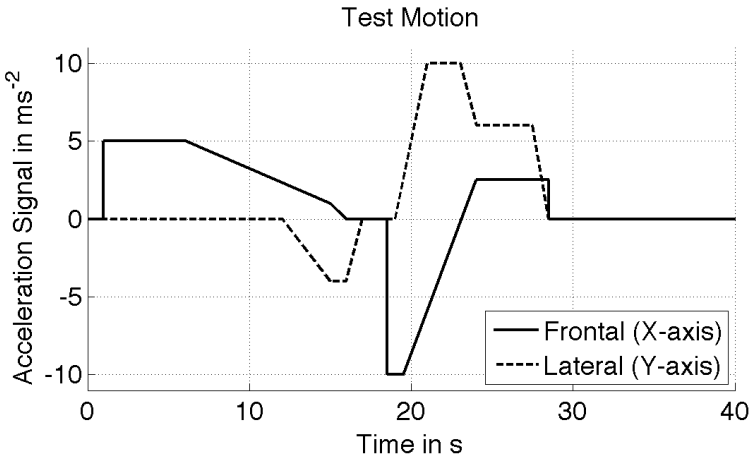
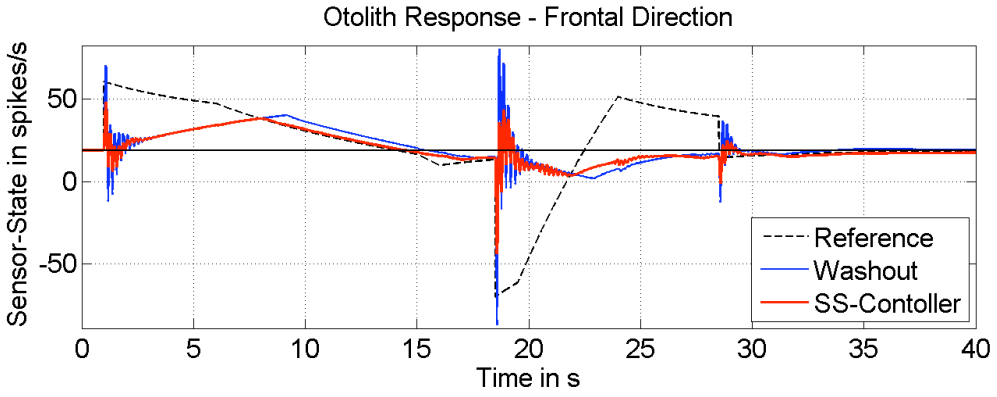
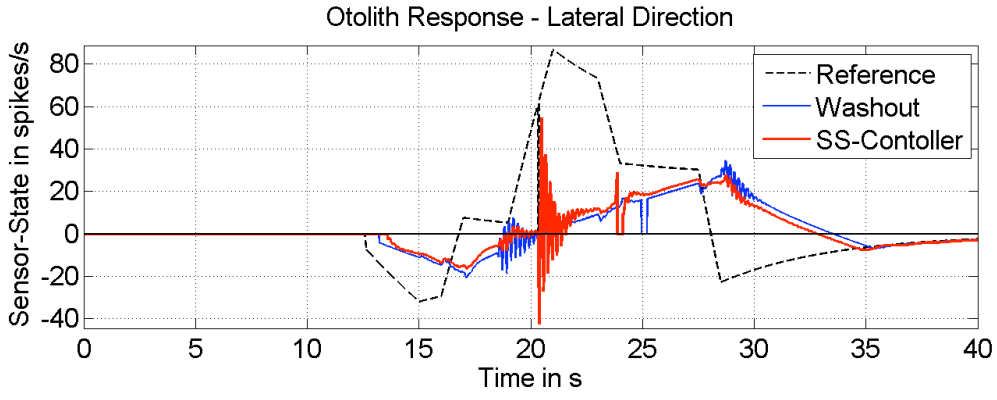


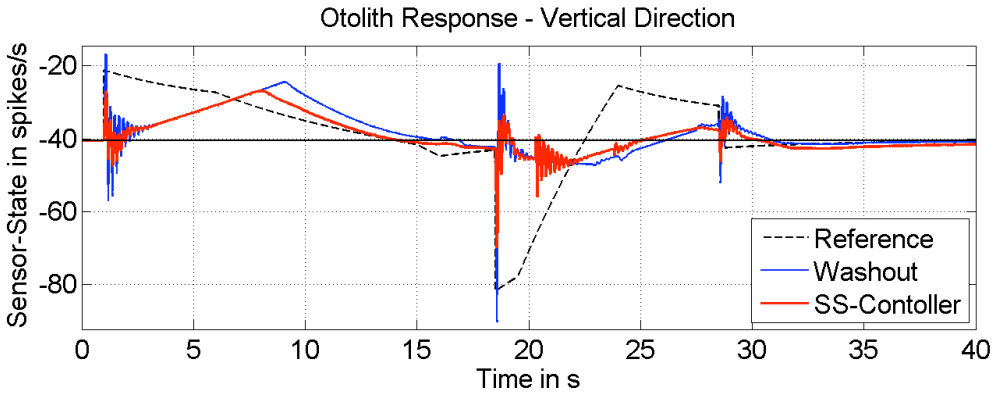
Figure 3.13 – Combined test motion

(a)



(b)





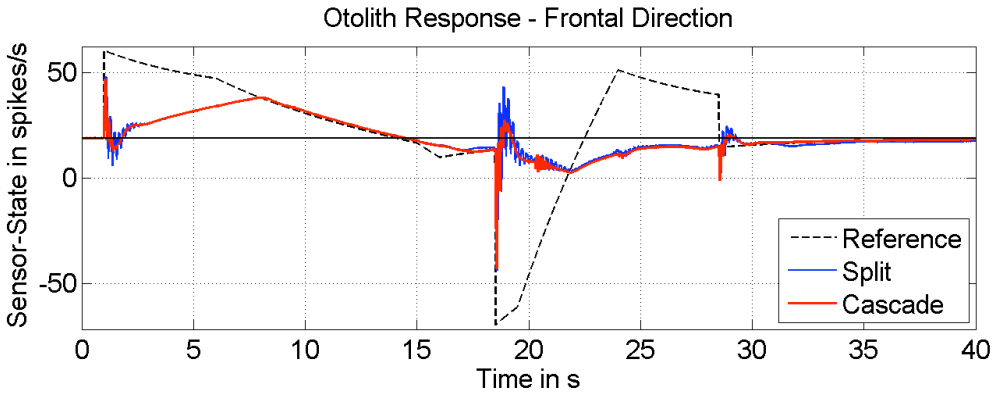
(c)

Figure 3.14 – Otolith response to combined motion
Comparing the SSC and the CWA

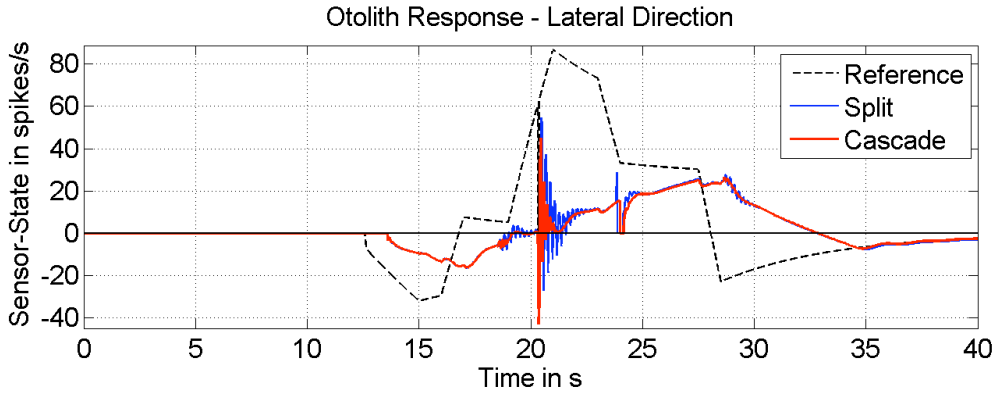
Comparing Motion Cueing System – SSC with Split vs. Cascade Loops

Finally, since both the quantitative error indicators and the qualitative evaluation of the sensor-states showed the superiority of the SSC over the CWA, two different configurations of the former were tested. The new structure integrates the motion cue generation and motion control problems by arranging them in cascade – rather than in series, as was the case of the SSC and CWA in the previous case. Figure 1.1 and 1.2 show the difference between the two approaches.

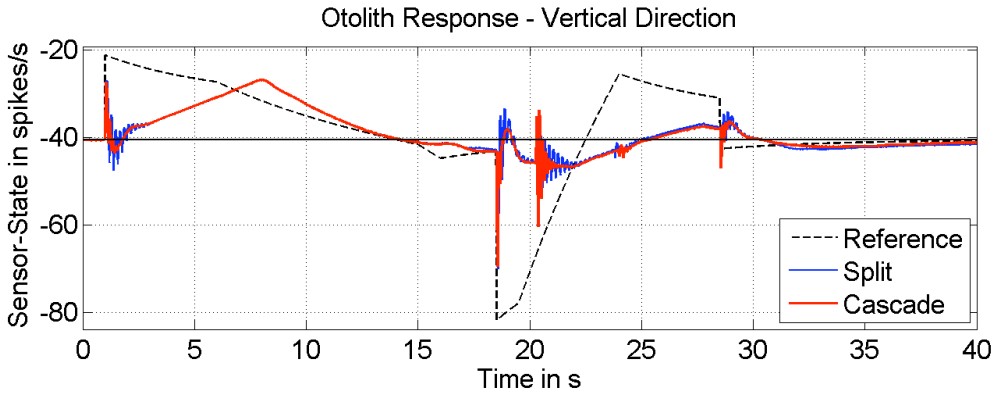
Figure 3.15 shows the relevant sensor-state graphs and the error indicators for this as well as the previous case are compiled in Table 3.2.



(a)



(b)



(c)

Figure 3.15 – Otolith response to combined motion
Comparing two configurations of the SSC

	Error Type	Wx	Wy	Wz	Sx	Sy	Sz	Cx	Cy	Cz
MAG	False	269	250	94	206	216	*65*	235	194	*66*
	Scaled	265	330	144	256	319	*131*	240	345	*136*
	Sum	534	580	238	462	*535*	*196*	474	*539*	*202*
VAR	False	699	422	403	348	414	207	162	246	168
	Scaled	1033	310	962	874	616	983	759	432	946
	Sum	1732	732	1365	1222	1030	1190	921	678	1114
			Classical Washout			S-S Conroller Split Loops			S-S Conroller Cascade Loops	

Table 3.2 – Simulation quantitative indicators comparison

In Table 3.2, the highlighted numbers show superior performance than the other two algorithms. As can be seen, the SSC in cascade loops concentrates the best performance indicators. None the less, the superiority of the SSC in split loops configuration over the CWA is also evident. One of the main advantages of implementing the sensor-state feedback is that it reduced the vibrations and jerkiness in the vestibular signal, as can be seen from Figures 3.14 and 3.15. If this is done in the cascade loops arrangement, this advantage becomes even clearer; and this becomes evident from the sensible reduction in the VAR indicators.

3.5.2 Experimental

The results obtained in simulation were validated by performing experiments on the real system at COLIVRI Laboratory. Of the three scenarios presented earlier, only the second one was replicated for experimental testing.

Scenario 1, where a motion system with a unitary transfer function is assumed, was aimed at evaluating the MDA independent from the motion system, thus it cannot be recreated in experiments. Scenario 3 is, so far, only exploratory. This implementation

of the Sensor-State Controller would require the solution of various problems beyond the scope of this work: estimating the kinematic state of the platform with precision, processing the signal in real-time and executing both the Cueing Algorithm and the motion control tasks in real-time.

From simulation to implementation

In order to implement the different motion cueing systems on the real test bed, various steps had to be taken due to the limitations in the available motion control system.

1. A model of the system was run in Matlab and Simulink. This provided the commanded lengths for each of the actuators as a function of time.
2. With these signals, another Matlab function wrote a text file with the code needed to execute the motion in YTerm (the control card's software).
3. The code was then downloaded to the control card with the YTerm software, so that the motion may be executed.
4. The different measurements were taken in two ways: servomotor information (position, error, torque) was taken from the control card by different subroutines in the YTerm code; inertial units were connected to the computer using a National Instruments Compact-DAQ and LabView software. The control card and LabView information was synchronized by programming digital outputs from the controller to the DAQ.
5. The signals are then processed in Matlab to obtain the sensor-states and quantitative error indicators necessary for motion cue evaluation.

In the future, a lower level programming language will be used in order to coordinate the motion cueing and motion control tasks. Perhaps in the form of a programmable motion controller or a different software built from scratch. When this is achieved in the near future, it would be possible to test the SSC in a cascade loops arrangement.

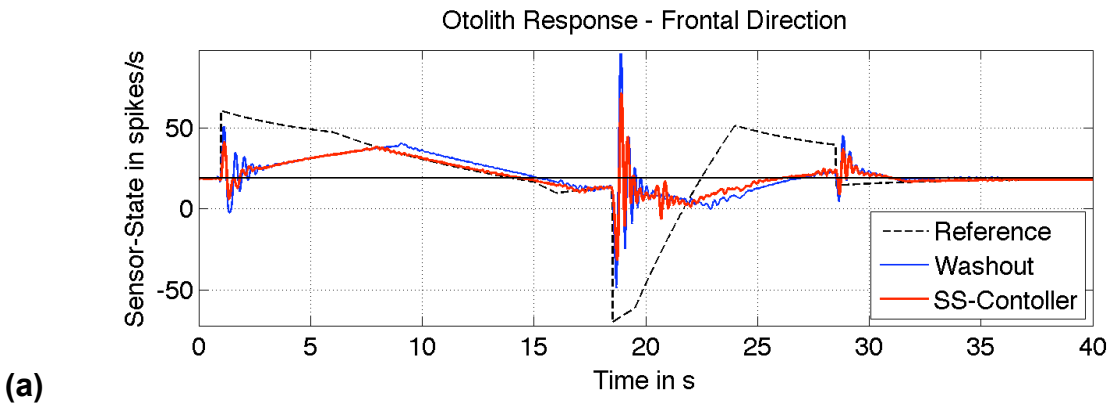
For this thesis, the validation of the CWA and split-loops-SSC were sufficient to fulfil the previously defined objectives. These results are shown in Figure 3.16. The sensor-states were estimated by processing the signal of the SEN inertial unit with Butterworth filters and consequently through the vestibular organs model.

These graphs show a similar behaviour of the otolith sensor-states to the one observed in simulation. Namely, the platform dynamics induces some vibrations in the sensor-state signal, but the SSC manages to reduce this undesired high-frequency content better than the CWA.

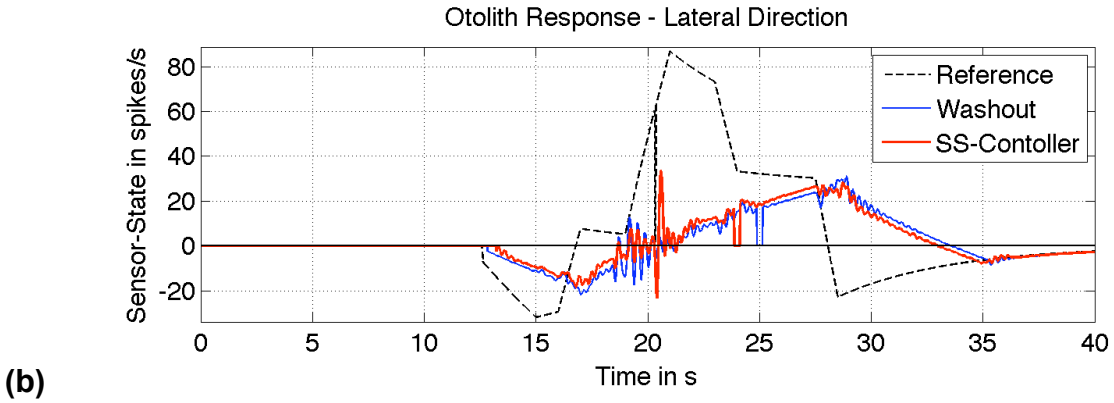
The quantitative error indicators of Table 3.3 further corroborate this.

	Error Type	Wx	Wy	Wz	Sx	Sy	Sz
MAG	False	235	252	96	178	215	66
	Scaled	*288*	*323*	143	*289*	*321*	133
	Sum	523	575	239	467	536	199
VAR	False	456	310	284	337	255	178
	Scaled	705	336	390	596	394	327
	Sum	1161	*646*	674	933	*649*	505
		Classical Washout			S-S Conroller Split Loops		

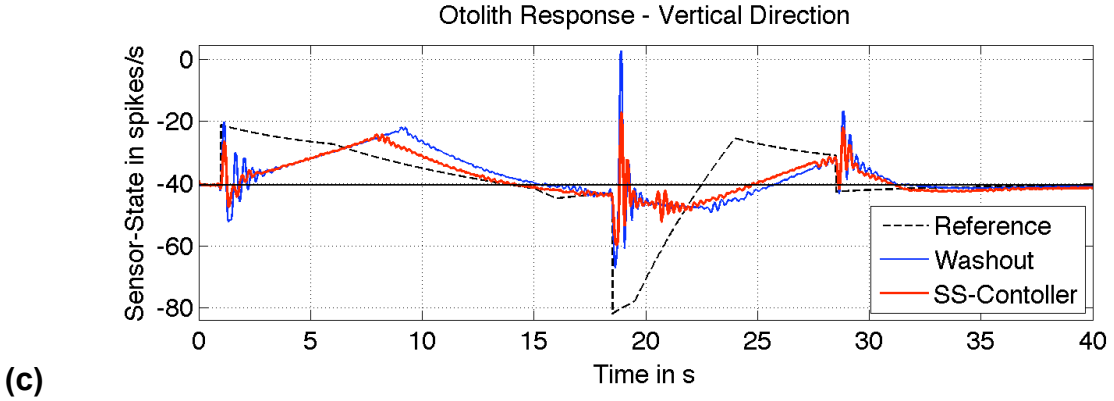
Table 3.3 – Experimental quantitative indicators comparison



(a)



(b)



(c)

Figure 3.16 – Otolith response to test motion comparison Testing the SSC and CWA on the real Stewart Platform

3.6 Discussion

The general conclusion drawn from Tables 3.1 to 3.3 is that the Sensor-State Controller would produce a better simulation of the test motions than the Washout. The graphical analysis could have been used to reach a similar determination, but the quantitative indicators make this statement objective. This marks an important advance in motion drive algorithm evaluation, which is more commonly assessed subjectively by expert pilots or simulator designers (Grant & Reid, 1997).

Even if it is important to analyze why each indicator may be larger or smaller for a certain algorithm during a particular maneuver, the direct comparison of the different values can lead to an unbiased and quick judgment of which algorithm is better for a particular maneuver.

The magnitude and variation indicators could also be computed using a 2-norm instead of the absolute-value integral, as shown in Eq. (3), as long as it is applied separately to false and scaled cues. In doing so, one would be weighing the error magnitude more than the time (or duration) of the error signal. In fact, any norm could be applied; a higher-level norm simply means a higher weight of the error magnitude relative to the time duration of the error.

$$MAG_2 = \left(\int_{t_0}^{t_f} \left[E(t) \right]^2 \cdot dt \right)^{1/2}$$

$$VAR_2 = \left(\int_{t_0}^{t_f} \left| \frac{dE(t)}{dt} \right|^2 \cdot dt \right)^{1/2}$$

One could also use a weighted sum of the false and scaled indicators in order to obtain one global value for each the MAG and the VAR in each axis, as shown in Eq. (4). The false cue would receive a higher weight, relative to the scaled cue (that is $\alpha, \beta > 1$), given that the latter is more harmful to motion perception aboard the simulator. The relative value of the weights could be assigned depending on the vehicle being simulated and the motion capabilities of the simulator.

$$MAG_{TOTAL} = \alpha \cdot MAG_{FALSE} + MAG_{SCALED}$$

$$VAR_{TOTAL} = \beta \cdot VAR_{FALSE} + VAR_{SCALED}$$

These possibilities give way to the broad application of the indicators to different simulator design scenarios.

Also worth mentioning is the noticeable similarity in the experimental results and the simulations that included a model of the Stewart Platform.

4. CONCLUSIONS AND FUTURE WORK

4.1 Motion Control

A new approach was taken to tune the motion controller gains. This approach allows the simplification of the system, while avoiding an inadequate loss of generality of the control problem.

This is an adequate approximation to the system because, even though the different actuators do interact to produce a certain motion on the moving platform, the control card is controlling each one on them independently.

The main advantage of using this uncertainty-based analysis is that its results can be more confidently extrapolated from the simple single-actuator system to the more complex, 6 degree-of-freedom platform.

The application of a robust performance method to tune the gains of the motion controller permitted a meaningful simplification of the system, with positive results on the performance of the Stewart Platform.

The extrapolation from the single actuator to the Stewart platform proved successful in practice; however, future work could be focused on a better understanding of the implications of this method through theoretical analysis.

This same robust approach could be taken with other controller structures such as feed forward or pseudo-differential PIDs.

Further work can also be carried out in the instrumentation and identification of the Stewart Platform.

4.2 Motion Cueing Algorithms

This project has successfully developed and tested a new motion drive algorithm based on the human-centered approach, both in simulation and experimentally.

Quantitative motion cue quality indicators were defined so that its performance could be compared to that of the standard motion drive algorithm used in the motion simulation industry.

The most relevant contribution of the indicators is that they link the motion cue directly to the quality of perceived motion aboard the simulator, and that they do so in a quantitative and objective way.

These indicators show how the feedback structure improved capabilities of the new algorithm to reduced motion cueing errors.

A flexible and powerful modelling tool for the whole simulator motion cueing system was developed in Simulink.

The dynamc behaviour of the model proved to be similar to that of the real system, making it a useful device for future work in motion simulation.

In the future, this motion drive algorithm and motion control design method can be implemented not only on simulators for passengers, but also pilots, where the system must react to the subject's input commands in real-time.

Another interesting line of work is to do test with naïve subjects in order to relate their opinions to the quantitative indicators. This would eventually facilitate the definition of rules to tune the motion drive algorithm parameters in terms of the results obtain with the indicators.

5. REFERENCES

Almonacid, M., Saltaren, R. J., Aracil, R., & Reinoso, O. (2003). Motion Plannig of a Climbing Parallel Robot. *IEEE Transactions and Robotics and Automation , 19* (3), 485-489.

Angelaki, D. E. (2005). The Physiology of the Peripheral Vestibular System: The Birth of a Field. *Journal of Neurophysiology , 93*, 3032-3033.

Borah, J., Young, L. R., & Curry, R. E. (1989). Optimal Estimator Model for Human Spatial Orientation. *Annals of the New York Academy of Sciences , 545* (1), 51-73.

Doyle, J., Francis, B., & Tannenbaum, A. (1990). *Feedback Control Theory.* New York, New York: Macmillan Publishing.

Elloumi, H., Bordier, M., & Maïzi, N. (2005). An Optimal Control Scheme for a Driving Simulator. *International Conference on Informatics in Control, Robotics and Automation.* Barcelona, Spain: INSTICC.

Exlar Corporation. (2008). *Exlar I Series Linear Actuator.* Electronic Catalogue, Minneapolis, MN.

Goldberg, J. M., & Fernández, C. (1976). Physiology of Peripheral Neurons Innervating Otolith Organs of the Squirrel Monkey. I, II & III. *Journal of Neurophysiology , 39* (4), 970-1080.

Goldberg, J. M., & Fernández, C. (1971). Physiology of Peripheral Neurons Innervating Semicircular Canals of the Squirrel Monkey. I, II & III. *Journal of Neurophysiology , 34* (4), 635-684.

Grant, P. R., & Reid, L. D. (1997). Motion Washout Filter Tuning: Rules and Requirements. *Journal of Aircraft , 34* (2), 145-151.

Holly, J. E., & McCollum, G. (1996). The Shape of Self-Motion Perception I & II. *Neuroscience , 70* (2), 461-513.

Ifediba, M. A., Rajguru, S. M., Hullar, T. E., & Rabbit, R. D. (2007). The Role of 3-Canal Biomechanics in Angular Motion Transduction by the Human Vestibular Labyrinth. *Annals of Biomechanical Engineering , 35* (7), 1247-1263.

Kane, T. R., & Levinson, D. A. (1985). *Dynamics: Theory and Applications.* USA: McGraw Hill.

Kihl, M., Herring, D., Wolf, P., McVey, S., & Kovuru, V. (2006). *Snowplow Simulator Training Evaluation.* Arizona State University, College of Design. Tempe: Arizona Department of Transportation.

Nanua, P., Waldron, K. J., & Murthy, V. (1990). Direct Kinematic Solution of a Stewart Platform. *IEEE Transactions on Robotics and Automation , 6* (4), 438-444.

Ochoa Lleras, N., & Rodriguez, C. F. (2009). Quantitative Indicators of Simulator Motion Cue Quality. *Dynamic Systems and Control Conference.* Hollywood, CA: ASME.

Ormsby, C. C., & Young, L. R. (1977). Integration of Semicircular Canal and Otolith Information for Multisensory Orientation Stimuli. *Mathematical Biosciences , 34*, 1-21.

Pouliot, N. A., & Gosselin, C. M. (1998). Motion Simulation Capabilities of Three-Degree-of-Freedom Flight Simulators. *Journal of Aircraft , 35* (1), 9-17.

Rabbit, R. D., Damiano, E. R., & Grant, J. W. (2004). Biomechanics of the Semicircular Canals and Otolith Organs. In S. M. Highstein, R. R. Fay, & A. N. Popper, *The Vestibular System* (pp. 153-201). New York: Springer.

Rodriguez, C. F., & Ochoa Lleras, N. (2008). Motion Simulation Based on Human Vestibular Sensors. *Dynamic Systems and Control Conference.* Ann Arbor, MI: ASME.

Rodriguez, C. F., & Ochoa, N. (2008). Motion Simulation Based on Human Vestibular Sensors. *Dynamic Systems and Control Conference.* Ann Arbor, MI: ASME.

Sciaviccio, L., & Siciliano, B. (1996). *Modelling and Control of Robot Manipulators* (Second ed.). England: Springer.

Smith, N., & Wendlandt, J. (2002). Creating a Stewart Platform Model Using SimMechanics. *MATLAB Digest , 10* (5).

Telban, F. J., & Cardullo, F. M. (2005). *Motion Cueing Algorithm Development: Human-Centered Linear and Nonlinear Approaches.* Langley Research Center. Hampton, Virginia: NASA.

Tsai, L.-W. (1999). *Robot Analysis. The mechanics of serial and parallel manipulators.* Canada: John Wiley & Sons Inc.

Tseng, H.-L., & Fong, I.-K. (2000). Implementation of a Driving Simulator Based on a Stewart Platform and Computer Graphics Technologies. *Asian Journal of Control , 2* (2), 88-100.

Yaskawa Electric America, Inc. (2009). *Sigma II Servo System Product Catalogue Supplement.* Waukegan, IL.

Yaskawa Electric America, Inc. (2004). *SMC-4000 User's Manual.* Waukegan, IL.