NATIONAL UNIVERSITY OF SCIENCE AND TECHNOLOGY POLITEHNICA BUCHAREST

DOCTORAL SCHOOL OF BIOTECHNICAL SYSTEMS ENGINEERING

PHD THESIS SUMMARY

Contributions to motion cueing algorithms for flight simulators

Author: Eng. Alina-Ioana CHIRA

PhD supervisor: Prof. dr. eng. Ion STROE

DOCTORAL COMMITTEE

President	Prof dr. eng. Andrei CRAIFALEANU	from	National University of Science and Technology POLITEHNICA Bucharest
PhD supervisor	Prof. dr. eng. CS I Ion STROE	from	National University of Science and Technology POLITEHNICA Bucharest
Reviewer	CS I dr. eng. Cătălin NAE	from	National Institute for Aerospace Research "Elie Carafoli" - INCAS
Reviewer	CS I dr. eng. Constantin OLIVOTTO	from	Academia Tehnică Militară Ferdinand I
Reviewer	Prof. dr. eng. Adrian-Mihail STOICA	from	National University of Science and Technology POLITEHNICA Bucharest

Bucharest
2024

Table of contents

1. Introduction

The PhD thesis proposes the development of a motion cueing algorithm for a flight simulator based on a serial robot motion platform using genetic algorithms for the optimization of the optimal washout filter based on linear quadric regulator (LQR). The design, implementation, testing and validation of the algorithm were performed using the dynamic simulation system for pilot-in-the-loop applications, RoFSim, developed at the National Institute for Aerospace Research and Development "Elie Carafoli" - INCAS. The proposed motion cueing algorithm converts aircraft dynamics into robot movements, generating high-fidelity motion within the physical limits of the flight simulator. This reproduction includes specific forces and accelerations, allowing the simulation to replicate sensations similar to those experienced in a real aircraft.

The proposed research topic is essential because creating a realistic flight simulation environment for existing and developing aircraft is necessary. This environment validates flight scenarios, incorporates new technologies, and provides training for pilots to handle situations that may arise during aircraft operation.

The obtained results demonstrate the stability and good performance of the implemented optimal washout filter, highlighting the potential use of the proposed algorithm in R&D projects and its further development.

Keywords: *flight simulator, motion cueing algorithm, serial robot, optimal washout filter, LQR, genetic algorithm, vestibular system*

1.1. Context

Flight simulation has long been a vital tool in the aerospace industry. In order to minimize the risks and costs of aircraft development ([1], [2], [3], [4]) flight simulation environments have been developed with the aim of improving flight quality [5], while maintaining a constant level of operational safety. Researchers make extensive use of flight simulators to gain knowledge in areas such as training procedures, understanding pilot/aircraft interface limitations, and ergonomics. Fixed-platform and motion-based simulators, parallel robots and serial robots have been developed in this context.

Over the past decades, the payload of commercial six-axis independent robotic systems has systematically increased. The first flight simulators based on serial robotic systems, developed for research purposes, were the Diamond DA42 at the German Aerospace Centre (DLR) ([6], [7]) and the Cyber-Motion Simulator at the Max Planck Institute (MPI) [8].

One effort in this direction is the dynamic simulation system for pilot-in-the-loop applications called RoFSim ([9], [10], [11]), a flight simulator using a serial robot as a motion platform. This simulator has been developed at the National Institute for Aerospace Research and Development "Elie Carafoli" - INCAS. The simulator provides motion pointing capabilities in addition to the basic functions of any flight simulator. While the hardware performance of the flight simulator motion system has advanced significantly, the development of the motion indication algorithm that translates simulated aircraft dynamics into actionable motion commands is slightly lagging. This PhD thesis formulates, designs, develops, and implements an optimal motion indication algorithm that converts aircraft dynamics into robot

motions. The algorithm generates high-fidelity motions within the physical limits of the flight simulator, reproducing specific forces and accelerations. This simulation aims to provide similar sensations to those experienced in a real aircraft cockpit.

1.2. Motivation

As the dynamic simulation system for pilot-in-the-loop applications is a simulator used for research purposes ([11]), this thesis aims to design an optimal motion cueing algorithm. The algorithm should be able to transform the real motion of a flight simulator maneuver into a logical and understandable motion for the serial robot workspace.

1.3. Thesis objectives

The PhD thesis aims to enhance the dynamic simulation system for pilot-in-the-loop applications by increasing its realism and ease of use. To achieve this, an optimal motion cueing algorithm has been developed to improve the flight simulation environment. This algorithm is useful in stability analyses of new aircraft conFiguretions [12], based on electric-hybrid architectures.

1.4. Thesis structure

The PhD thesis comprises seven chapters.

Chapter one introduces flight simulators and motion indication algorithms for various motion platforms, including parallel and serial robotic platforms. The chapter also outlines the objectives and motivation of this study.

The second chapter presents the current state of research on mathematical modelling of motion cueing algorithms specific to flight simulators. The aim is to convey to the pilot the acceleration sensations they would experience in a real aircraft.

Chapter three covers the mathematical modelling of the dynamic simulation system for pilot-in-the-loop applications. The chapter begins with an introduction of the simulator platform, followed by a presentation of the simulator modules and functional structure. The subsequent section presents the reference systems and coordinate transformations used in this work. The chapter covers the mathematical modelling of the serial robotic system. It analyses and simulates the direct kinematics and inverse kinematics of the robotic system.

Chapter four presents the procedure for scaling the workspace of the flight simulator, starting from the singularities handling method and local optimization. The conclusions of the chapter provide the minimum and maximum joint limits of the motion platform for the test conFiguretion of the motion indication algorithm.

Chapter five covers the mathematical modelling of the motion cueing algorithm. The chapter begins with an introduction to the optimal washout filter based on LQR, followed by a presentation of the algorithm's general structure. The following section presents the scaling and liminting module of the algorithm, which ensures the safe operation of the ABB IRB 7600-500 serial robotic system during flight simulator testing. The subsequent sub-chapter briefly introduces the mathematical model of the vestibular system, a crucial component of the motion cueing algorithm. The chapter presents the implementation procedure of the motion cueing

algorithm based on the optimal washout filter. The chapter concludes with the optimized algorithm model being used with evolutionary genetic algorithms.

Chapter six presents the flight scenarios that were used to test the motion cueing algorithm and the corresponding results.

The thesis concludes with a chapter of general conclusions, original contributions, and future research perspectives in the field of flight simulator-specific motion indication algorithms. Additionally, the bibliography used in carrying out the research work in the thesis is also presented.

2. State-of-the-art on mathematical modelling of flight simulator motion cueing algorithms

Chapter 2 provides an overview of the current state of the art in the development of motion cueing algorithms. This thesis addresses the problem of transmitting motion cues in the flight simulator using motion cueing algorithms, also known as washout filter algorithms.

2.1. Introduction to motion cueing algorithms for flight simulators

The washout filter algorithm aims to simulate the acceleration sensations a pilot would experience in a real aircraft on a motion platform with limited space ([13], [14], [15]). It uses the linear accelerations and angular velocities of the simulated aircraft as input and outputs the trajectory to be followed by the end-effector of the motion platform in a Cartesian coordinate system. This algorithm is composed of three channels (rotational, translational, and tilt coordination), each of which consists of low-pass or high-pass filters, depending on the behaviour that the flight simulator is intended to represent.

The most important theories in this area are the optimal ([16], [17], [18], [19], [20]), adaptive ([15], [19], [21], [22], [23], [24]) and robust ([16], [18], [19], [25], [26], [27]) washout filter.

2.2. Classical washout filter algorithm

The classical washout filter algorithm is a widely used basic solution in various types of simulators due to its simplicity and ease of tuning ([28], [29], [30], [31]). It has several advantages, including short processing time and stable performance. Research has demonstrated that the classical washout filter algorithm has limitations ([32], [33], [34]), ([35], [36], [13]). It is inflexible as it necessitates a tuning process that concentrates on the worst-case scenario, resulting in conservative motion and poor workspace utilization. Therefore, the algorithm may not be suitable for certain circumstances. The washout filter algorithm has a main disadvantage in that it does not consider human perception, as it ignores the vestibular system in its structure.

2.3. Optimal washout filter algorithm

The optimal washout filter algorithm is based on human motion sensation and considers the vestibular system ([37], [38]). The method integrates a mathematical model of the human vestibular system to minimize the sensation error between the pilot in the simulator and the

pilot in a real aircraft cockpit. The method utilises optimal control techniques based on quadratic linear regulators to develop higher-order filters for real-time application. A cost function is designed that depends on the sensation error between real and simulated aircraft pilots, as well as the platform motion.

When developing an optimal washout filter based on LQR, the main challenge is to identify an appropriate constraint matrix of linear transfer functions, $W(s)$. This matrix connects the simulator and vehicle motion inputs to minimize the cost function by restricting both the error in perception between the simulator and the aircraft pilot and the movement of the platform should be considered. Figure 2.1 shows the diagram of the optimal washout filter based on LQR.

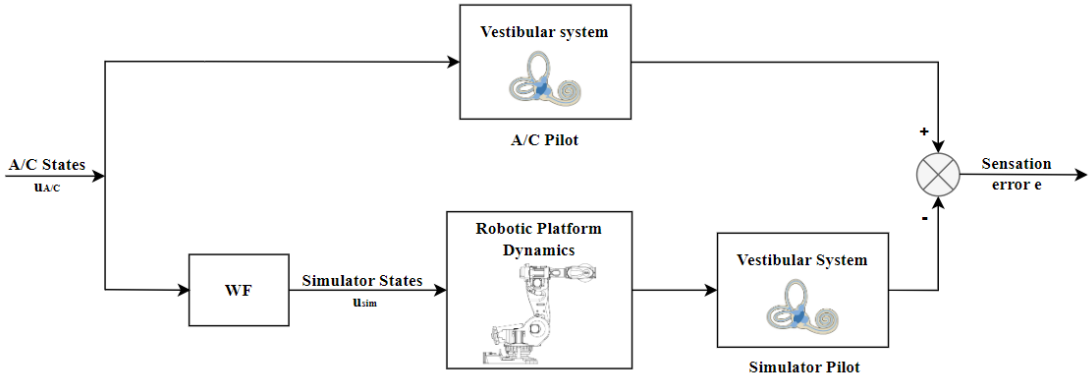


Figure 2.1 Structure of the optimal washout filter based on LQR, adaptation [39]

2.4. Centralisation of motion cueing algorithms characteristics

Four types of motion cueing algorithms were identified for the design, development, implementation, and testing of a motion indication algorithm for flight simulators based on the state-of-the-art research in the literature.

- Classical washout filter;
- Adaptive washout filter;
- Optimal washout filter;
- Robust optimal washout filter.

The main advantages of the classical washout filter are [40]:

- The model is both mathematically and computationally simple, making it cost-effective;
- It is also relatively easy to design, although experience is required to modify it based on feedback from pilots during experimental testing of the algorithm in the simulator.

Among the disadvantages of the classical washout filter algorithm, the following can be mentioned:

- the use of mainly linear elements, thus not fully exploiting the capabilities of the simulator nor considering the non-linear characteristics of human motion perception (the human body is sensitive to both linear specific force and angular velocity [16], an experience that can be simulated by implementing a vestibular system model);

- the design must prioritize the most critical states due to fixed parameters that may result in minimal motion during lighter manoeuvres, a limitation that can be addressed by more sophisticated implementations of the washout filter.
- The adaptive washout filter (AWF) was proposed by Parrish and co-workers [15] along with Reid and co-workers [41]. It has adaptive washout filter amplitudes that vary to minimize a cost function that penalizes the motion error, the magnitude of the motion and the change in adaptive parameters from their initial values [42]. It is important to note that only the high-pass filter amplitude is adaptive, while the low-pass filter amplitude remains fixed. The adjustment of a constraint filter involves selecting cost weights and adaptive filter amplitudes.

The method has several advantages: [43]:

- the filter generates reduced false motion indices compared to the classical washout filter;
- choosing cut-off frequencies, damping values, and initial amplitude for this filter presents the same difficulties as the classical washout filter. However, the choice of cost weights is more intuitive for less experienced users because they are directly related to acceleration and angular velocity errors, instead of being related to cut-off frequencies that do not explicitly refer to any motion error;
- the motion platform's adaptive features provide more realistic motion cues when the simulator is close to neutral and reduce motion fidelity only when the simulator approaches its physical limits. This allows for better utilization of the platform's capabilities;
- the cost function to be minimized can be varied using non-quadratic functions, such as those introduced in [40], which may or may not include vestibular models.

The adaptive washout filter algorithm has several drawbacks, including:

- computationally compared to the classic washout filter is heavier;
- the cut-off frequencies, damping and initial values of the filter amplitude are adjusted based on the accelerations in the most critical case;
- minimises the motion errors (Cartesian accelerations and angular velocities) between the aircraft and the robotic platform.

To consider the error of perception as a variable to be minimized, Sivan and co-workers [14] proposed an optimal washout filter (OWF). The resulting washout filter was obtained from a linear quadratic regulator (LQR) that minimizes a cost function that takes into account the error of perception, and, linear displacement from the initial position and angular displacements and speed, as well as movement controls of the platform. The advantages of this filter are as follows:

- minimizes the pilot's sensation error instead of the actual motion error by incorporating a vestibular model;
- takes into account the correlation coefficient, considered in the optimal constraint filter scheme [16], and, as a form tracking criterion, helps to generate signals that can track reference signals more accurately;
- ease of adjustment by a user with little experience, as it adapts by adjusting the cost function weights related to more intuitive variables.

The disadvantages of this filter are presented below:

- the optimal control scheme produces fixed parameter filters, similar to the classical washout filter scheme, which do not fully exploit the motion capabilities of the motion platform and need to be adjusted for the most critical maneuvers;
- the tilt speed limit is not included in this algorithm because it had a negative effect on the algorithm behaviour [41];
- system constraints are not explicitly considered [16].

Table 2.1 provides a summary of the characteristics of existing constraint filter schemes based on the research conducted in this chapter, considering the advantages and disadvantages of each filter in terms of computational time, difficulty of algorithm tuning, inclusion of the human vestibular system model, and increased motion platform workspace utilisation. At the same time, the table shows the type of robotic system for which the motion detection algorithm has been designed, implemented and tested so far, i.e. parallel robotic systems (denoted by SRP) and serial robotic systems (denoted by SRS). The green colour represents an advantage and the grey colour represents a disadvantage.

Table 2.1 Characteristics of motion cueing algorithms

	CWF	**AWF**	**OWF**	**RWF**	**CWF/Q**
Reduced computational time					
Easy algorithm tuning					
Human perception error					
Workspace – uniform quality					
Implementation of robotic systems	SRP / SRS	SRP	SRP	SRP	SRP / SRS

3. Mathematical modelling of a dynamic simulation system for pilot-in-the-loop applications

3.1. Introduction – overview of the dynamic simulation system

The dynamic simulation system for pilot-in-the-loop applications platform (Figure 3.1) aims to achieve a complex interconnection of systems that model flight simulation by integrating the flight simulator model based on a serial robotic platform. The aim is to create a closed-loop flight simulator with 6 degrees of freedom by using the robotic arm as a motion platform.

The entire proposed system consists of an ABB IRB 7600-500/2.55 industrial robot manipulator arm with six independent axes, at the end of which an aircraft cabin cell is mounted. The assembly moves along a GUDEL TMF-4 V2 track with a travel of one metre. The cell mounted at the end of the robotic arm provides the kinematic redundancy required to cope with the constraints of robotic actuation. The purpose of the airframe is to create a cockpit environment, equipped with specific avionics equipment, a virtual reality-based visual system and an integrated audio system to provide the pilot with as many cues as possible to mimic what happens in a real cockpit. At the same time, a cell-robot connection interface is used to enable data to be transferred from the cockpit to the ground (monitoring and control room) and vice versa, creating two-way communication between the two environments.

The main objective of the proposed conFiguretion is to utilise the flight simulator platform as a versatile simulation environment for a wide range of flight scenarios and to increase the fidelity of the simulation environment for dynamic stability analysis of new aircraft conFiguretions such as hybrid electric architecture.

Figure 3.1 Dynamic simulation system for pilot-in-the-loop applications, Credits: INCAS

The implementation of this motion simulator platform aims to conduct studies to test motion perception, human-machine interaction, human psycho-physiology and also to evaluate pilot control behaviour in basic experimental missions.

The aircraft simulation structure for the robotic platform-based motion system is shown in Figure 3.2. The inputs given by the operator (pilot) are passed to the aircraft dynamic model, generating the aircraft state vector. Passing the aircraft state vector through the motion indication algorithm produces the desired motion cues and the states of the robotic platform providing the motion to the simulator. The desired robot platform states are then transformed into the axis workspace, generating commands to the six robot axes.

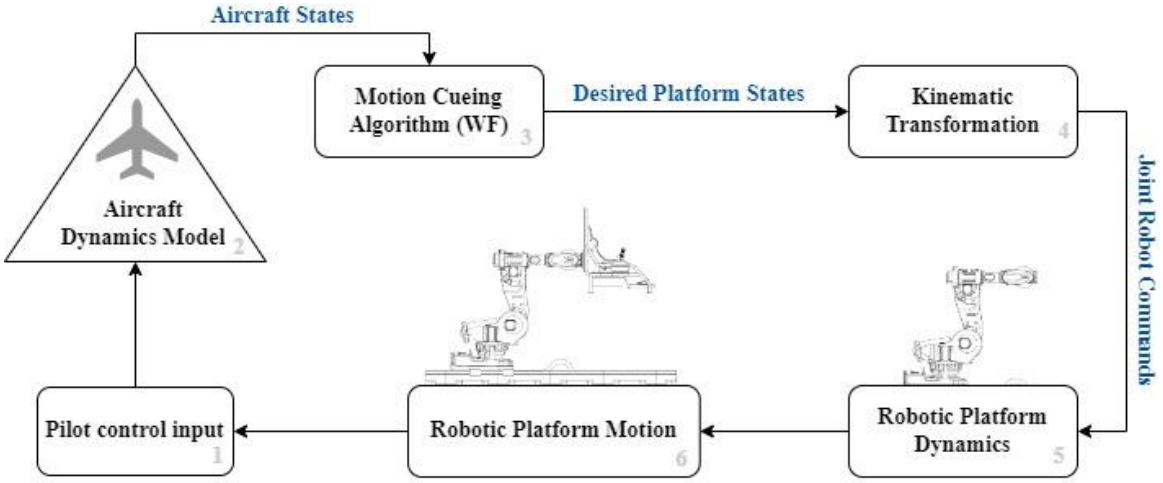


Figure 3.2 Aircraft simulation structure diagram

Inputs provided by the operator (pilot) are passed to the aircraft dynamics model, generating the aircraft state vector. Passing the aircraft state vector through the motion algorithm produces the desired motion cues and the robot platform states that provide the motion to the simulator. The desired robot platform states are then transformed from the degrees of freedom space to the axis workspace, generating commands to the six robot axes. The axis motion commands act as inputs to the robot platform, resulting in the actual motion of the simulator. The operator (pilot) inputs are transferred to the dynamic simulation model running on a real-time platform. The operator (pilot) signals can come from various specific control devices such as flap position, rudder actuation, throttle actuation, etc. The outputs from the dynamic aircraft model are measured in terms of translational accelerations, rotational angles and angular velocities; outputs that are required to reproduce the motion feedback to the simulator. The aim is to induce realistic motion in the simulation cabin. Obviously, due to the limitations of the simulator workspace, the motion cannot be identical to that of the aircraft and must be compressed. After determining the position and orientation of the cabin, the individual command input from the six-degree-of-freedom robot is calculated from the inverse kinematics transformation. The resulting command inputs are passed through the collision prevention filter and are sent to the robot. This step closes the motion feedback loop for the test pilot.

The flight simulator, based on the six-axes serial robotic system, involves the development of the simulator components as an open system composed of modifiable and scalable modules. The six degrees of freedom allow the simulation of aircraft rotations (roll, pitch and yaw) as well as Cartesian coordinate displacements (X, Y, Z).

The main components of the aircraft model dedicated to the flight simulator system are shown in Figure 3.3.

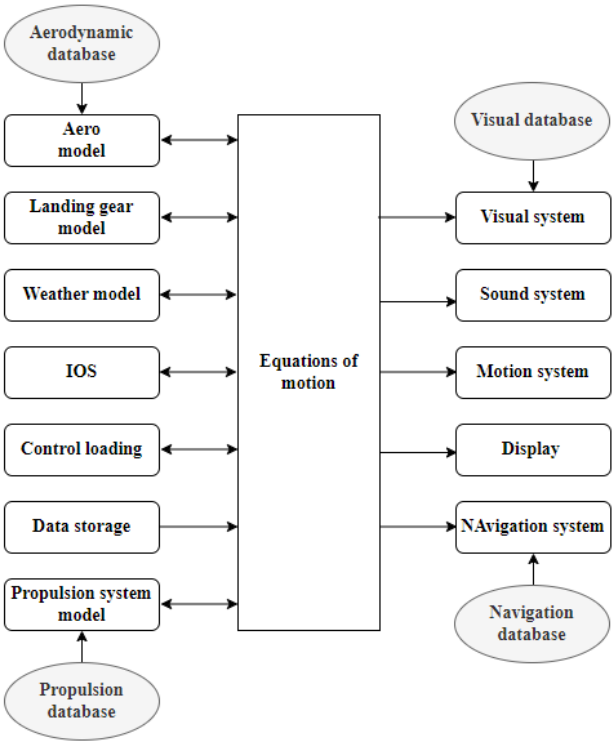


Figure 3.3. Flight simulator model components

The dynamic simulation system for pilot-in-the-loop applications based on the motion platform covers simulation in three areas:

- Aircraft dynamics simulation (aerodynamic flight model and engine, aircraft systems, ground handling, flight controls, cabin conFiguretion).
- Environment simulation (ATC environment, environment, terrain and airfield, navigation).
- Motion simulation (motion cues, visual cues, audio cues).

3.2. Coordinate systems

Several reference systems are used in the definition and implementation of the motion cueing algorithm. These reference systems are defined in this chapter and shown in Figure 3.4.

3.2.1. Aircraft reference system

Aircraft reference system $SR_{A/C}$ is derived from the aircraft centre of gravity, denoted by CG. The reference system $SR_{A/C}$ has an orientation for X_{CG}, Y_{CG} and Z_{CG}, which is parallel to the simulator reference system SR_{sim} and the pilot aircraft reference system SR_{AP}.

3.2.2. Simulator reference system

The simulator reference system SR_{sim} originates at the centre of the robot joint 6 to which the simulator is attached, i.e. at the centre of the counterpart of the robot simulator airframe attachment. X_{sim}, faces forwards and și Z_{sim} upwards relative to the simulator cockpit, and

Y_{sim} is directed to the right side of the pilot. The $X-Y$ plane is parallel to the floor of the simulator cockpit.

3.2.3. Pilot-aircraft reference system

The pilot-aircraft reference system SR_{AP} originates from the same relative cockpit position as the simulator reference system SR_{sim}. SR_{AP} has the same orientation for X_{AP}, Y_{AP} and Z_{AP} in relation to the cockpit as the simulator reference system SR_{sim}.

3.2.4. Pilot-simulator reference system

The pilot-simulator reference system SR_{SP} originates from the same relative position of the cockpit as the simulator reference system SR_{sim}. SR_{SP} has the orientation for X_{SP} identical to the orientation X_{sim}, Y_{SP} is directed to the right side of the pilot, and Z_{SP} is oriented downwards in relation to the cockpit.

3.2.5. Inertial reference system

The inertial reference system SR_I is fixed by the Earth, with Z_I ligned with the gravity vector g. Its origin is located in the centre of the base of the motion platform. X_I is pointing forward, and Y_I to the right side in relation to the simulator pilot.
Figure 3.4 shows the vectors that define the relative position of the reference system. R_{sim} defines the location of SR_{SP} relative to SR_{sim}. Similarly, $R_{A/C}$ defines the location of the reference system SR_{AP} relative to $SR_{A/C}$.

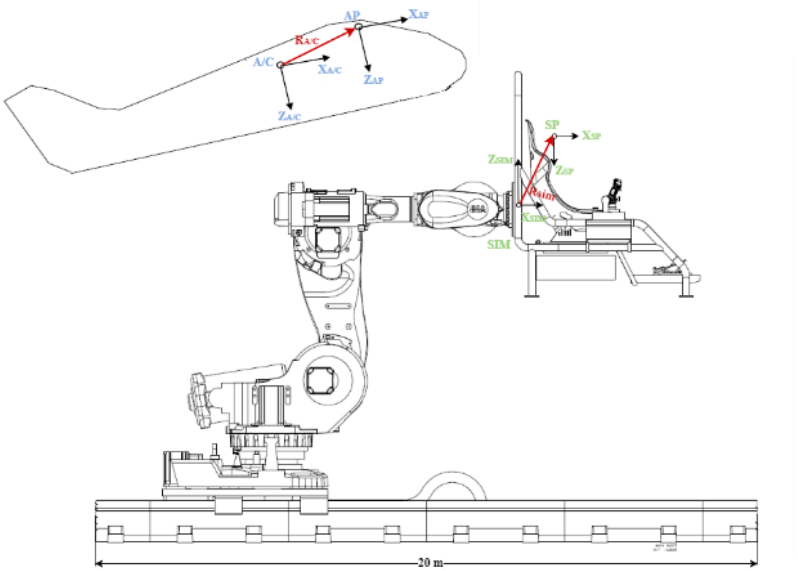


Figure 3.4 Reference systems positioning

3.3. Coordinate transformations

The orientation between the aircraft reference system $SR_{A/C}$ and the simulator reference system SR_{sim} is achieved by using a direction cosine matrix through ZYX rotation.

$$\boldsymbol{SR_{A/C} = DCM_{ZYX} \cdot SR_{sim}} \quad 3.1$$

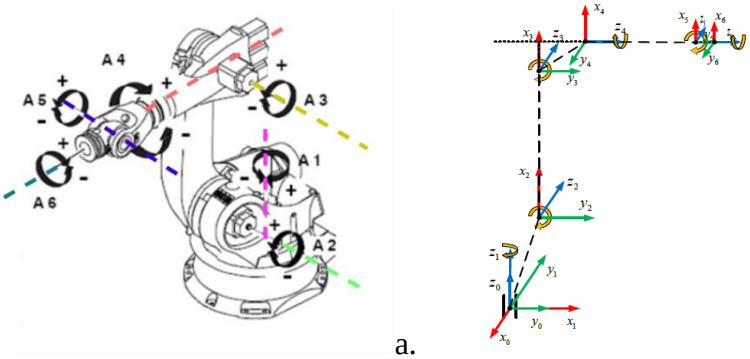
where

$$\boldsymbol{DCM} = \begin{bmatrix} c(\theta)\cdot c(\psi) & c\,(\theta)\cdot s\,(\psi) & -s(\theta) \\ s(\phi)\cdot s(\theta)\cdot c(\psi) - c(\phi)\cdot s(\psi) & s(\phi)\cdot s(\theta)\cdot s(\psi) + c(\phi)\cdot c(\psi) & s(\phi)\cdot c(\theta) \\ c(\phi)\cdot s(\theta)\cdot c(\psi) + s(\phi)\cdot s(\psi) & c(\phi)\cdot s(\theta)\cdot s(\psi) - s(\phi)\cdot c(\psi) & c(\phi)\cdot c(\theta) \end{bmatrix} \quad 3.2$$

3.4. **Mathematical modelling of serial robotic system**

In this subchapter, we present the inverse kinematics module for the serial robotic system, which appears in the general flight simulator operating scheme as a link module between the specific motion display algorithm module and the specific robot platform dynamics module.

Figure 3.5 shows the kinematic structure of the ABB IRB 7600-500 robot. Figure 3.5a shows the robot actuators placed at joints between the serial drive chain elements, and Figure 3.5b shows the reference systems specific to each other joint of the robot.



a. b.

Figure 3.5 IRB 7600-500 serial robot struture (a. Robot axes, b. Robot axis reference systems)

Robot kinematics is divided into direct and inverse kinematics.
The correlation between direct kinematics and inverse kinematics is shown in Figure 3.6 [44].

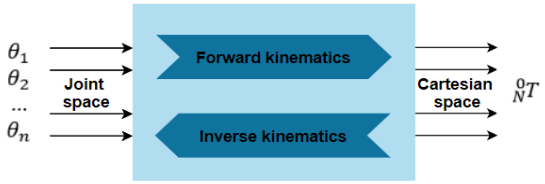


Figure 3.6 Schematic description of the direct and inverse kinematics of the serial robot

3.4.1. **Direct kinematics of the serial robot**

The Denavit-Hartenberg (D-H) mathematical representation [45] is used to study the direct kinematics of the serial robot, which shows that a global transformation between two joints requires four parameters describing the kinematics of the robot. Denavit-Hartenberg (D-H) gives the joint transformation between two joints, considering element lengths and joint angles.

The following two tools were used to model the ABB IRB 7600-500/2.55 robot MATLAB R2022b Robotics System Toolbox [46] and Robotics Vision Control Toolbox [47], both following the Denavit and Hartenberg (D-H) terminology.

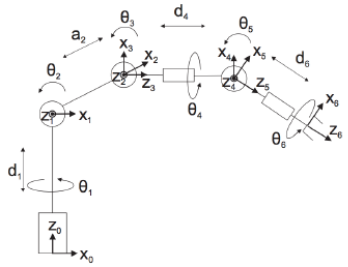


Figure 3.7 Robot diagram IRB 7600-500

According to the defined coordinate system, the corresponding element parameters are defined as follows:

a_{i-1} $[mm]$: distance measured from z_{i-1} to z_i along x_{i-1};

α_{i-1} [°]: angle of rotation from z_{i-1} to z_i around the axis x_{i-1};

d_i $[mm]$: distance measured from x_{i-1} to x_i along z_i;

θ_i [°]: angle of rotation from x_{i-1} to x_i around the axis z_i.

These four parameters describe the kinematic chains of the robot. The Denavit -Hartenberg (D-H) parameters for the IRB 7600-500/2.55 robot are presented in Tablee 3.1. The D-H parameters were calculated using robotic system-specific input data such as element size (Figure 3.8), joint rotation/displacement direction (Figure 3.5a) and the basic robot reference system (Figure 3.5b).

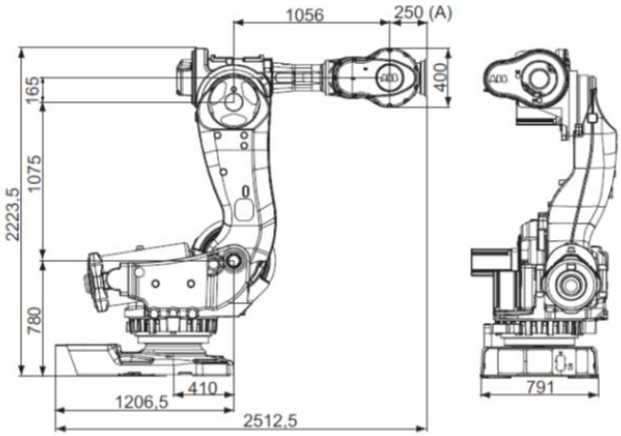


Figure 3.8 ABB IRB 7600-500/2.55 robot size specification [48]

Table 3.1 ABB IRB 7600-500/2.55 robotic system D-H parameters and joint limits

Link element i	**a_{i-1} $[mm]$**	**α_{i-1} [°/rad]**	**d_i $[mm]$**	**θ_i [°/rad]**	**Min/max limit joint [°/rad]**	**Robot joints**
1	0.41	-90	0.78	0	- 180 / +180	q_1
		$-\pi/2$			$-\pi$ / π	
2	1.075	0	0	-90	- 60 / +85	q_2
				$-\pi/2$	$-\pi/3/\ \pi/3$	

3	0.165	-90	0	0	- 180 / +60	q_3
		$-\pi/2$			$-\pi$ / $\pi/3$	
4	0	90	1.056	0	- 300 / +300	q_4
		$\pi/2$			$-5\pi/3$ / $5\pi/3$	
5	0	-90	0	0	- 100 / +100	q_5
		$\pi/2$			$-5\pi/9$ / $5\pi/9$	
6	0	0	0.25	180	- 360 / +360	q_6
				π	-2π / 2π	

The direct kinematics of the robot calculates the position and attitude of the last actuator, the end-effector (EE), taking into account the values of the six joints. The general formula for the transformation of the element ${}^{i-1}_{i}T$ in the kinematics of the robot arm is:

$$ {}^{i-1}_{i}T = T_x(\alpha_{i-1})T_x(a_{i-1})T_z(\theta_i)T_z(d_i) \tag{3.3} $$

where ${}^{i-1}_{i}T$ is a homogeneous transformation matrix, α_{i-1} și a_{i-1} are translation and rotation around the axis x_{i-1}, respectively d_i și θ_i re translation and rotation around the axis z_{i-1}.
The direct kinematics of the manipulator is expressed by:

$$ \mathbf{{}^{0}_{1}T(q_1){}^{1}_{2}T(q_2){}^{2}_{3}T(q_3){}^{3}_{4}T(q_4){}^{4}_{5}T(q_5){}^{5}_{6}T(q_6) = T_f} \tag{3.4} $$

3.4.2. Inverse kinematics of the serial robot

Solution of inverse robot kinematics - means that the value of the robot joint variables is determined according to the given position of the end effector and the attitude of the manipulator [49]. The method of solving the inverse transformation consists of multiplying inversely a given transformation on both sides of the robot kinematics equation at the same time. The specific formula is expressed as follows:

$$ {}^{0}_{i}T^{-1}(q_i){}^{0}_{6}T = {}^{i}_{i+1}T(q_{i+1}){}^{i+1}_{i+2}T(q_{i+2}){}^{i+2}_{i+3}T(q_{i+3})\dots{}^{5}_{6}T(q_6) \tag{3.5} $$

Since the position of the end element ${}^{0}_{6}T$ is known, one can obtain the product of the inverse of each transformation on the left-hand side of the equation and the transformation on the right-hand side of the equation.

$$ {}^{0}_{6}T = {}^{0}_{1}T(q_1){}^{1}_{2}T(q_2){}^{2}_{3}T(q_3){}^{3}_{4}T(q_4){}^{4}_{5}T(q_5){}^{5}_{6}T(q_6) \tag{3.6} $$

To first find q_1 in terms of the known elements, calculate the inversion of the link transformation which are pre-multiplied as follows ${}^{0}_{1}T(q_1)$:

$$ {}^{0}_{1}T^{-1}(q_1){}^{0}_{6}T = {}^{0}_{1}T^{-1}(q_1){}^{0}_{1}T(q_1){}^{1}_{2}T(q_2){}^{2}_{3}T(q_3){}^{3}_{4}T(q_4){}^{4}_{5}T(q_5){}^{5}_{6}T(q_6) \tag{3.7} $$

where ${}^{0}_{1}T^{-1}(q_1){}^{0}_{1}T(q_1) = I$, and I is the identity matrix.
In the end there will be 12 simultaneous sets of nonlinear equations to solve. The 12 elements of the nonlinear matrix on the right-hand side are either zero, constant or functions from q_2 to q_6.

The inverse kinematics problem of the 6 DoF serial robot [44] was solved using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) optimisation algorithm and the inverse kinematics solution was simulated using two different simulation environments: MATLAB R2022b and Robot Studio [50]. The first simulation process uses the Matlab Robotics System Toolbox [46] to check the end-effector position and determine the joint angles for each set of points in the proposed trajectory. The second simulation process uses the RAPID program [51] from Robot Studio [50], to validate the results recorded in the first simulation using the same algorithm. The trajectory and motion of the robot, as well as joint angle tracking, are described in the RAPID programming language, which writes the data for analysis. The determination of the position and attitude of the end effector was performed using special functions in the Robot Studio program (CalcRobT and CalcJointT). The Robot Studio functions find the trajectory defined by five different points, and the position of the end-effector for these five points is shown in Table 3.3. The last three columns in Table 3.3 show the deviations expressed in $[mm]$.

Table 3.2 ABB IRB 7600-500 - End position for a specified path

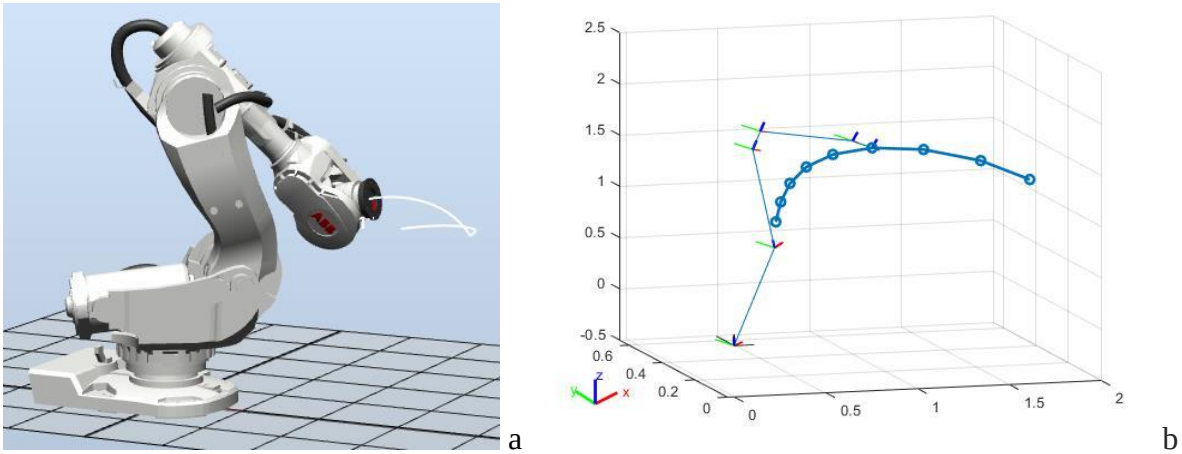
Point	**Trajectory points [mm]**			**End effector position [mm]**			**Deviation [mm]**		
	x	**y**	**z**	**x**	**y**	**z**	**Δx**	**Δy**	**Δz**
1	1900.0	100.0	1133.3	1899.85	100.227	1133.52	0.1500	-0.2270	-0.2200
2	1838.4	332.4	1128.3	1838.60	331.704	1128.32	-0.2000	0.6960	-0.0200
3	1677.8	502.8	1113.2	1678.12	502.442	1113.18	-0.3200	0.3580	-0.0200
4	1469.9	585.0	1088.7	1470.18	584.900	1088.67	-0.2800	0.1000	0.0300
5	1258.9	587.8	1055.3	1258.90	587.800	1055.30	0	0	0

For each of the points 1, 3 and 5 in the trajectory, the values of the joint angles (joint angle C) were calculated by inverse kinematics and compared with those obtained by simulating the motion of the ABB IRB 7600-500 series robot using Robot Studio with the RAPID program (joint angle R). The results in the table below show that there are small differences between the values of the joint angles.

Table 3.3 Values of manipulator joint angles

Point 1			**Point 3**			**Point 5**		
Joint Angle C	**Joint Angle R**	**Δ_Joint Angle**	**Joint Angle C**	**Joint Angle R**	**Δ_Joint Angle**	**Joint Angle C**	**Joint Angle R**	**Δ_Joint Angle**
3.4759	3.4682	0.0077	19.3838	19.3997	-0.0159	30.2200	30.2257	-0.0057
21.4093	21.4218	-0.0125	6.0040	15.9947	0.0093	2.5363	2.5247	0.0116
24.7475	24.7419	0.0056	33.8021	33.8135	-0.0114	54.6952	54.7075	-0.0123
4.8087	4.8028	0.0060	24.7384	24.7495	-0.0111	34.7130	34.7169	-0.0039
-46.2650	-46.2643	-0.0007	-52.4983	-52.5038	0.0055	-62.1140	-62.1179	0.0039
-3.3290	-3.3244	-0.0046	-15.6684	-15.6748	0.0064	-17.9533	-17.9535	0.0002

Graphical results of the predefined trajectory in MATLAB and Robot Studio environments for inverse kinematic simulations presented in the paper [44] are highlighted in Figure 3.9.



a b

Figure 3.9 Simulated robot trajectory IRB7600-500 (a. RobotStudio, b. MATLAB)

In this sub-section, an analytical solution of the direct and inverse kinematics of the ABB IRB 7600-500 series robot has been proposed. The accuracy of the models in the simulation environment has been tested for efficient use. To obtain the solution, a geometric approach was considered by using geometric relationships between different manipulator elements. The inverse kinematics model was simulated in two different environments and it was observed that the end effector moved along a predefined trajectory, confirming the effectiveness of the model. The purpose of this analysis is to validate the inverse kinematics model developed in the MATLAB simulation environment, which is part of the program developed in the same simulation environment that serves as an offline test platform for the motion display algorithm. A graphical representation of the joint position as a function of time was also developed. This study was carried out to provide a basis for improving aerospace robotics problems when used as motion platforms for flight simulators.

4. Scaling the workspace for the dynamic simulation system

To ensure a secure and stable workspace for the six degrees of freedom simulator, based on the IRB 7600-500 serial robotic platform, it was necessary to perform the workspace scaling procedure. This involved defining hardware and software limitations.

To ensure safety, the pilot was only able to operate the simulator within limited intervals of robot movement. This prevented collisions with other objects in the working area. However, these modifications resulted in additional constraints and a reduction in the robot's working area. When comparing the working area of the IRB 7600-500/2.55 robot with the working area of the robot coupled with the simulator cabin, it is evident that the simulator's working area is smaller due to the size of the simulator cell.

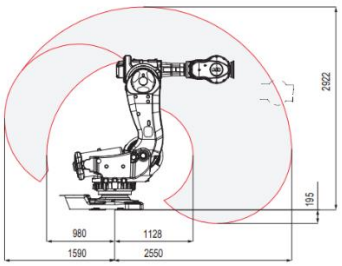


Figure 4.1 Robot workspace IRB 7600 -500 [48]

Different workspace constraints are defined for the flight simulator depending on its application. It is important to note that there is a dependency between the angles of the robot joints, meaning that limiting one joint will affect the others. Therefore, independent maximization of each joint angle range is impossible. As a result, the following optimization problem was formulated:

$$max\ \vec{\Omega} \cdot \vec{f_c}(\vec{q})\ ,\ \vec{q}\ \in\ R^6 \tag{4.1}$$

with

$$\vec{f_c}(\vec{q}) = \{\Delta X, \Delta Y, \Delta Z, \Delta\phi, \Delta\theta, \Delta\psi\}^T \tag{4.2}$$

where X, Y, Z are the Cartesian positions, ϕ, θ, ψ the rotation angles (roll, pitch, yaw), and $\vec{\Omega} \in R^6$ the weighting factor vector representing the joint constraint to control the robot motion within the allowed joint angles.

4.1. Hardware joint angle limitations

The manufacturer specifies the robot joint angle intervals in the data sheet. These hardware limitations are accompanied by software limitations imposed by the manufacturer. These limitations are triggered before reaching the hardware limit as a safety measure.
The range of the robot joint conFiguretion is highlighted.

$$\vec{q} \in [\vec{q}_{min}, \vec{q}_{max}] \tag{4.3}$$

Table 4.1 Defining the limits of the IRB 7600-500 robot

IRB 7600-500/2.55 robot	**Axis**	**Min joint limit** $\vec{q}_{min}$	**Max joint limit** $\vec{q}_{max}$	**Speed limit**	**Acceleration limits**
A	q_1	- 180 [°]	+ 180 [°]	75 [°/s]	-
B	q_2	-60 [°]	+85 [°]	50 [°/s]	-
C	q_3	-180 [°]	+60 [°]	55 [°/s]	-
D	q_4	- 300 [°]	+ 300 [°]	100 [°/s]	-
E	q_5	- 100 [°]	+100 [°]	100 [°/s]	-
F	q_6	- 360 [°]	+ 360 [°]	160 [°/s]	-
Track	q_7	-10 [mm]	20010 [mm]	90 [m/min]	86 [m/s^2]

4.2. Collision avoidance

The design of the simulator cell significantly affects the definition of joint ranges of motion. To define the collision avoidance condition, we considered cockpit size, flange position, flange angle, and subject (pilot) size. It is crucial to guarantee the system's safety throughout the robot's entire range of motion. Therefore, we must ensure that there are no collisions in the context of different motion combinations. The objects considered were the floor, the Gudel

TMF-4 track, the IRB 4600 robot, the IRC5 controller of the IRB 4600 robot, the IRC5 controller of the IRB 7600 robot, and the wall near the track.

$$V(object_i, \vec{q}) \cap V(object_k, \vec{q}) = \emptyset \quad 4.4$$

where $object_i = \{cell\ simulator, robot\ IRB\ 7600, pilot, \dots\}$,
and $object_{\mathrm{k}} = \{robot\ IRB\ 4600, track\ Gudel\ TMF-4, controler\ IRC5, wall, \dots\}$.

4.3. Software limitation of joint angles

Different types of constraints and workspace limitations need to be applied depending on the simulation type. After conducting offline simulations, several software limitations were established. These include the installation of SafeMove mode, software implementations in the IRC5 controller, and limitations of the q_5 axis to avoid singularity errors.

The optimization-based trajectory generation problem is formulated using the singularity handling method and modified to meet the requirements of the trajectory planning algorithm.

Either

$$x_e = f(q) \quad 4.5$$

direct kinematics of the robot arm, where the effector position is denoted by x_e, joint positions are denoted by q.
Inverse kinematics is represented as follows:

$$q = f^{-1}(x_e) \quad 4.6$$

It is implied that there are multiple common trajectories $q(t)$ that lead to the same effector trajectory $x_e(t)$. One method for computing the desired inverse kinematics for serial robots is to minimize the error between the reference and effector positions while considering constraints, with the robot joint positions as optimization variables [52].

$$min\|x_{eref} - x_{eq}\| \quad 4.7$$

where x_{eref} is the calculated reference position. As illustrated in Figure 4.2, the trajectory $q^i = q(t_i)$ s computed as a result of an optimization at each sampling step.

Near singularities, the dynamical limits of the system and the stability of the solution are guaranteed by constraints, but for cases of mechanical impossibility a positioning error is introduced. This approach computes the locally optimal motion of the robotic system to perform the tasks given by the optimization criteria.

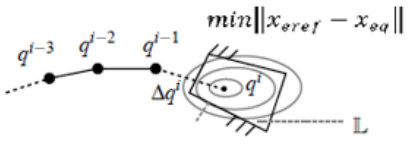


Figure 4.2 Local optimisation

Figure 4.2 shows the use of local optimization to determine the next joint angle at each time step. The optimization problem $\min\|x_{eref} - x_{eq}\|$ is solved by considering the robot's dynamic con straints, which define a space of possible solutions.

To consider the dynamic behaviour of a system in the optimisation problem, the differential equations are reformulated as optimisation constraints.

The use of a serial robotic conFiguretion as a motion platform for flight simulators presents challenges in terms of trajectory planning. Maintaining the desired trajectory requires very high angular velocities of the joints in near-singular conFiguretions. If the planned trajectory approaches a singularity, the actuation speeds' limits may cause trajectory errors, leading to a failure stop by the robot control.

Considering the specific constraints on joint angles and space requirements, we specify the joint angle ranges q_3, q_4 și q_5 as side conditions for the optimization problem. These ranges have a significant impact on the workspace.

To reduce the six-dimensional optimization problem to a two-dimensional one, we generate and utilize conFiguretion spaces:

$$\max A\,(\vec{q})\,,\ \vec{q}\ \in\ R^2 \tag{4.8}$$

with rectangular surface

$$D(\vec{q}) = (q_{4max} - q_{4min}) * (q_{5max} - q_{5min}) \tag{4.9}$$

and $D(\vec{q}) \in S$, $S \cong$ numerous combinations of joint angles q_4 and q_5 that can be achieved without any collisions.

The optimization problem is not affected by the joint angle q_1 due to the robot's design, which assumes the absence of any additional obstacles within its rotationally symmetric workspace.

All possible combinations of joint angles q_2, q_3, q_4 și q_5 are checked for collisions with a resolution of 10 degrees. Once the desired ranges of joint angles are defined q_4 and q_5, the conFiguretion space of q_2 and q_3 can be generated quickly.
The optimisation process yielded the following ranges for joint angles:

Table 4.2 Limitations of the joint angle range for optimised conFiguretion of the IRB 7600-500 robot

IRB 7600-500/2.55 Robot	**Joint**	**Min joint limit** $\vec{q}_{min}$	**Max joint limit** $\vec{q}_{max}$	**Min joint limit op** $\vec{q}_{min,o}$	**Max joint limit op** $\vec{q}_{max,o}$
A	q_1	- 180 [°]	+ 180 [°]	- 180 [°]	+ 180 [°]
B	q_2	-60 [°]	+85 [°]	-60 [°]	+85 [°]
C	q_3	-180 [°]	+60 [°]	-180 [°]	+40 [°]
D	q_4	- 300 [°]	+ 300 [°]	- 175 [°]	+ 175 [°]
E	q_5	- 100 [°]	+100 [°]	- 90 [°]	+90 [°]
F	q_6	- 360 [°]	+ 360 [°]	- 360 [°]	+ 360 [°]

Optimising joint angle margins can be challenging because limiting one joint angle can significantly impact the limits of other joint angles.

The secondary conditions for joint angles q_3, q_4 and q_5 were derived based on the application-specific constraints that the simulator must meet. Joint angle q_1 was not relevant

to this investigation. These secondary conditions reduced the six-dimensional optimization problem to a two-dimensional one. The collision-free Cartesian workspace was investigated by varying the joint angle ranges of q_4 and q_5 within their conFiguretion spaces.

5. Mathematical modelling of the motion cueiong algorithm

The purpose of motion cueing algorithms (MCA) is to produce motion cues that lead to human perception. A motion cueing algorithm uses tilt coordination to reproduce the effect of translational acceleration motion supported by the gravity vector to mimic human perception detected by the vestibular system.

5.1. Getting started

The washout filter is a motion algorithm that transforms the acceleration and angular velocity of a simulated vehicle into the motion of the simulator platform. The motion produced must be within the limits, constraints and restrictions of the motion platform used for the flight simulator, as presented in the previous chapter.
The optimal washout filter used is based on four assumptions:

- the vestibular system dominates the perception of motion cues in a flight simulator;
- the discrepancy between the motion cues between the real aircraft and the motion simulator can be measured by the root mean square of the vestibular error;
- real aircraft motion can be modelled as a random process independent of motion platform constraints;
- dynamic systems, including vestibular systems, can be represented by linearised equations.

The algorithm structure shown in Figure 2.1 contains two separate channels that generate the perception of motion between a real aircraft and a flight simulator. In both channels, the vestibular system represents the pilot's vestibular system. Ideally, the simulator output is identical to the input.

The purpose of this algorithm is to determine a transfer function $W(s)$ consisting of low-pass and high-pass filters. $W(s)$ filters the aircraft input $u_{A/C}$ to obtain the simulator input u_{sim} to minimise the cost function, which includes the pilot's error of perception.

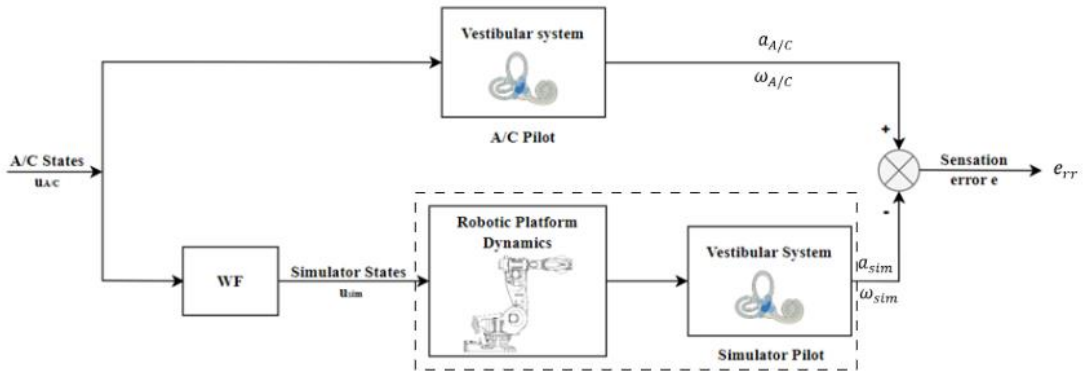


Figure 5.1 General structure of the optimal washout algorithm, adaptation [39]

Figure 5.1 shows the optimal control technique. The objective function of this algorithm includes the perception error and the constraints related to the motion in the workspace of the flight simulator motion platform, where $a_{A/C}$ and a_{sim} represent the detected acceleration at the actual position and at the workspace position. Also, $\omega_{A/C}$ and ω_{sim} are the rotational velocities detected at these positions. Due to the use of a model with two inputs and two outputs, four transfer functions (W_{11}, W_{21}, W_{12} și W_{22},), are used, each representing the effect of each input on each output. In order to take into account the structure and parameters of the motion algorithm, it is necessary to convert the actual input ($u_{A/C}$) and the flight simulator input (u_{sim}). According to Figure 5.1, there are two different ways of comparing the perception of the real motion with that in the flight simulator.

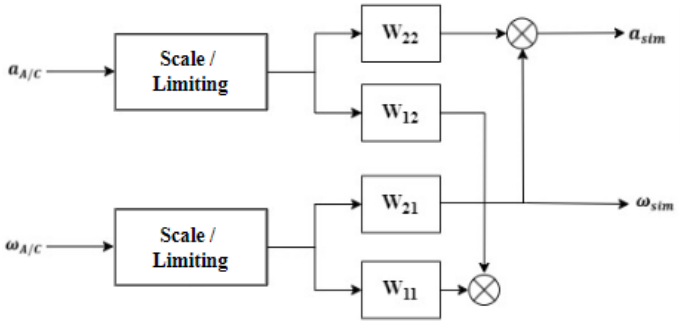


Figure 5.2 Optimal washout filter algorithm, adapted from [17]

5.2. Scaling and limiting

The simulator's motion platform has physical limits, so its dynamics must be constrained within these limits and constraints. Therefore, within the working scheme of the motion algorithm, a constraint has been applied to the translational and rotational channels, and at the same time, a scaling has been applied to the model input signals.

Limiting and scaling apply to both the aircraft translational input signals $\boldsymbol{a_{A/C}}$and the rotational input signals $\boldsymbol{\omega_{A/C}}$. Limiting and scaling change the amplitude of the input signal uniformly across all frequencies. Limiting is a non-linear process that reduces the signal so that it is limited to a value less than a certain amplitude. Limiting and scaling can be used to reduce the motion response of a flight simulator.

A third-order polynomial was used for scaling and was implemented in the general scheme of the motion algorithm. When the magnitude of the input to the simulator motion system is small, it is desired that the amplitude be relatively high, otherwise the output will be below the pilot's perception threshold. And when the input magnitude is high, it is desired that the amplitude is relatively small to avoid the case where the simulator attempting to exceed the hardware limits.

For the implementation of the procedure, the input was denoted by $\boldsymbol{x_a}$, and the output by $\boldsymbol{x_z}$. Then $\boldsymbol{x_{amax}}$ was defined as the desired maximum input and $\boldsymbol{x_{zmax}}$ as the maximum output, and $\boldsymbol{s_0}$ and $\boldsymbol{s_1}$ the slopes at $\boldsymbol{x_a = 0}$ and $\boldsymbol{x_a = x_{amax}}$ respectively.
Four desired non-linear scaling characteristics are defined as follows:

$$\begin{gathered} x_a = 0 \Longrightarrow x_z = 0 \\ x_a = x_{amax} \Longrightarrow x_z = x_{zmax} \\ x_z'|_{x_a=0} = s_0 \\ x_z'|_{x_a=x_{amax}} = s_1 \end{gathered} \qquad 5.1$$

The third-order polynomial scaling used to provide functions with desired characteristics is of the form:

$$x_z = p_3 {x_a}^3 + p_2 {x_a}^2 + p_1 {x_a}^1 + p_0 \quad 5.2$$

where

$$\begin{gathered} p_0 = 0 \\ p_1 = s_0 \\ p_2 = {x_{amax}}^{-2}(3 \cdot x_{zmax} - 2 \cdot s_0 \cdot x_{amax} - s_1 \cdot x_{amax}) \\ p_3 = {x_{amax}}^{-3}(s_0 \cdot x_{amax} - 2 \cdot x_{zmax} + s_1 \cdot x_{amax}) \end{gathered} \quad 5.3$$

Scalarea parametrilor pentru canalul translațional:

$$max|x_{amax}| = 6\ m/s^2 \quad 5.4$$

$$max|x_{zmax}| = 6\frac{m}{s^2}, \text{ pentru } X \quad 5.5$$

$$max|x_{zmax}| = 0.8\frac{m}{s^2}, \text{ pentru } Y \text{ și } Z \quad 5.6$$

and the coefficients are:

$$\begin{gathered} s_0 = 1, \text{ pentru } X \\ s_0 = 0.2, \text{ pentru } Y \text{ și } Z \\ s_1 = 0.1, \text{ pentru } X, Y \text{ și } Z \end{gathered} \quad 5.7$$

Parameter scaling for the rotational motion:

$$max|x_{amax}| = 3.14\frac{rad}{s} \quad 5.8$$

$$max|x_{zmax}| = 1.57\frac{rad}{s} \quad 5.9$$

and the coefficients are:

$$\begin{gathered} s_0 = 0.785, \text{ pentru } X, Y \text{ și } Z \\ s_1 = 0.1, \text{ pentru } X, Y \text{ și } Z \end{gathered} \quad 5.10$$

The linear accelerations on all axes are limited to $6\frac{m}{s^2}$ for the translational channel, and the limit for the rotational channel is $3.14\frac{rad}{s}$. To compute the polynomial scaling coefficients for each aircraft degree of freedom for input to the optimal motion algorithm, the flowchart shown in Figure 5.3 is implemented.

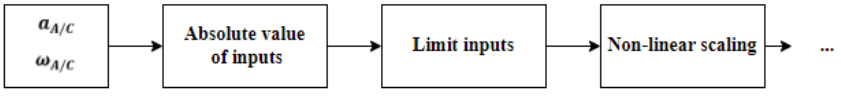


Figure 5.3 Flow chart for limiting and scaling

5.3. Mathematical model of the vestibular system

As shown in Figure 5.1 , the mathematical model that implements the motion cueing algorithm includes a module that contains the mathematical model of the vestibular system. This module is essential for the implementation of the optimal washout filter algorithm. For a good implementation of the mathematical models for the two elements of the vestibular system, the semicircular canal and the otolith organ, research on motion thresholds is carried out to define specific values to be used in the development of the motion cueing algorithm.

The vestibular system (Figure 5.4) is located in the inner ear and consists of semicircular canals that detect angular motion and otolith organs that detect linear motion [53].

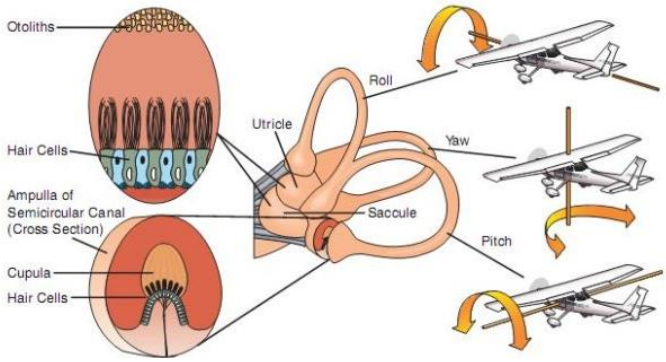


Figure 5.4 Relationship between the vestibular system and human perception [54]

The vestibular system comprises two degrees of freedom: overload and pitch, as illustrated in Figure 5.5, where a_{Ax} denotes linear acceleration, and $\dot{\theta}$ represents rotational velocity. The outputs from the vestibular model are a_{Ax_s}, which represent felt acceleration and $\dot{\theta}_s$, which represents felt rotational velocity. The vector g represents gravitational acceleration. The integral of rotational velocity with respect to displacement is defined by the sign of the integral.

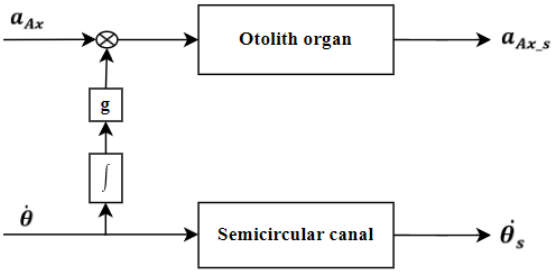


Figure 5.5 Model of the vestibular system for human perception

The vestibular system's semicircular canal and otolith organ were mathematically modelled using MATLAB/Simulink, as depicted in Figure 5.6. The models from Figure 5.1, Figure 5.2 and Figure 5.5 were integrated into the model.

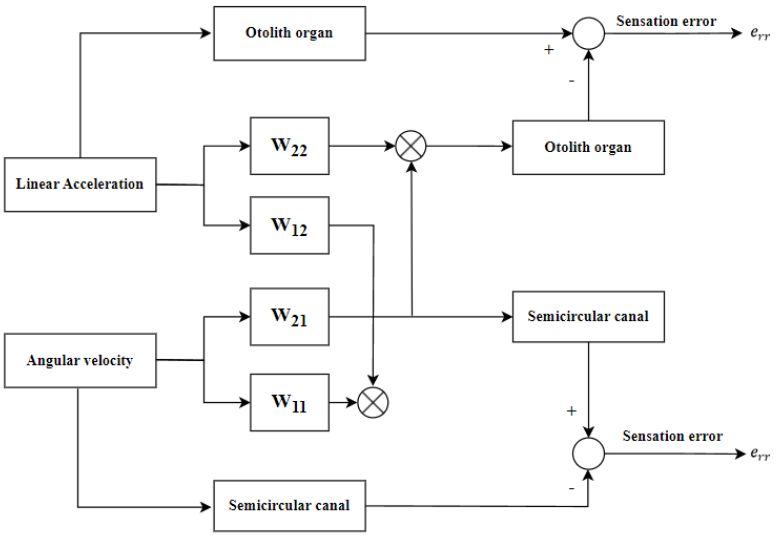


Figure 5.6 Model of vestibular system wint integrated washout filter

The goal of the optimal washout filter based on LQR (Figure 3) is to determine the transfer function $W(s)$ that relates the flight simulator motion input (u_{sim}) to the aircraft motion input ($u_{A/C}$).

$$u_{sim}(s) = W(s) \cdot u_{A/C}(s) \tag{5.11}$$

Control inputs, including accelerations and Euler angles, are used to generate the basic commands for the desired motion. Thus, the optimal washout filter algorithm determines the simulator acceleration by minimising the human feel error between the simulator and the aircraft, as well as the linear and angular motion of the platform. The objective of the algorithm is therefore to limit the human perception error and platform motion within the physical workspace of the robotic platform. Input *u* is represented as follows:

$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ a_{Ax} \end{bmatrix} \tag{5.12}$$

An important step in the implementation of the algorithm is how the definition of the tilt coordination effect is. The tilt coordination aims to calculate the low frequency components on the horizontal axis, X direction, and on the lateral axis, Y direction, while the low frequency components on the vertical axis, Z direction, are not calculated. Figure 5.7 shows how the tilt coordination is formulated to generate acceleration on the horizontal and lateral axes.

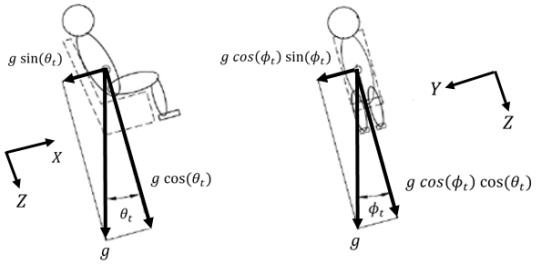


Figure 5.7 Formulation of the tilt coordination

The specific surge force in the centre motion of the simulator can be obtained as follows, but considering the approximation for small angles, sin (θ_t) and cos (θ_t) can be replaced by θ_t and 1 respectively:

$$f_s = a_{Ax} \cdot \cos(\theta_t) + g \cdot \sin(\theta_t) \cong a_{Ax} + g \cdot \theta_t \quad 5.13$$

The otolith organ detects linear movement in the inner ear. Meiry and co-workers [55] proposed a modified model for indicating linear acceleration and tilt based on the research in [37].

The specific detected value a_{Ax_s} is related to the specific stimulus strength detected by the otolith model and is entered into the equation below as follows:

$$a_{Ax\_s} = k_O \frac{s + A_0}{(s + B_0)(s + B_1)} \cdot f_s \quad 5.14$$

By applying the Laplace transform, a new form of the equation can be obtained. The term $\frac{1}{s}$ is the integration of the angular velocity.

$$f_s(s) = a_{Ax}(s) + g \cdot \frac{1}{s} \cdot \dot{\theta}(s) \quad 5.15$$

Substituting equati 5.13 into equation 5.14 gives the following equation:

$$a_{Ax\_s} = k_O \frac{s + A_0}{(s + B_0)(s + B_1)} \left(a_x(s) + g \cdot \frac{1}{s} \cdot \dot{\theta}(s) \right) \quad 5.16$$

where A_0, B_0 și B_1 are the specific parameters of the otolith model.

Equation 5.16 can be rewritten as follows:

$$\dddot{a}_{Ax\_s} + a \cdot \ddot{a}_{Ax_s} + b \cdot \dot{a}_{Ax_s} = c \cdot \dot{u}_1 + d \cdot u_1 + e \cdot \int u_1 dt + f \cdot \dot{u}_2 + h \cdot u_2 \quad 5.17$$

The representation in state space is given by the following form thus:

$$\begin{aligned} \dot{x}_{ot} &= A_{ot} \cdot x_{ot} + B_{ot} \cdot u \\ a_{Ax\_s} &= C_{ot} \cdot x_{ot} + D_{ot} \cdot u \end{aligned} \quad 5.18$$

where x_{ot} represent the states for the otolithic model and u represents the input to the vestibular system, and A_{ot}, B_{ot}, C_{ot} și D_{ot} are defined as follows:

$$A_{ot} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -b & -a & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -b & -a \end{bmatrix} \quad 5.19$$

$$B_{ot} = \begin{bmatrix} c & 0 \\ d - a \cdot c & 0 \\ e & 0 \\ 0 & f \\ 0 & h - a \cdot f \end{bmatrix} \quad 5.20$$

$$C_{ot} = [1 \quad 0 \quad 0 \quad 1 \quad 0] \quad 5.21$$

$$D_{ot} = [0 \quad 0] \quad 5.22$$

The angular velocity $\dot{\theta}_s$ detected by the semicircular system is related to the actual angular velocity. Semicircular channels detect the sensation of rotational motion. Telban and Cardullo [56] proposed a transfer function of the semicircular canals model to detect angular velocity sensation when applied as a stimulus. This transfer function includes the torsional pendulum model, the adaptation operator, and a lead term. This function offers the most accurate

approximation of the vestibular sensation system's true dynamics for rotational motion. The semicircular canal model is defined as follows:

$$\dot{\theta}_s = \frac{k_{sc} \cdot \tau_1 \cdot \tau_a \cdot s^2 (1 + \tau_L \cdot s)}{(\tau_1 \cdot s + 1) + (\tau_2 \cdot s + 1) + (\tau_a \cdot s + 1)} \dot{\theta} \quad 5.23$$

where τ_1, τ_2, τ_L și τ_a are the long time constant, short time constant, lead term and adaptation operator constant, respectively.

Equation 5.23 can be rewritten as follows:

$$\dot{\theta}_s = \frac{T_5 \cdot s^3 + T_4 \cdot s^2}{s^3 + T_3 \cdot s^2 + T_2 \cdot s + T_1} \dot{\theta} \quad 5.24$$

where $T_0, T_1, T_2,\ T_3$ $și$ T_4 are constants and can be defined as follows:

$$T_1 = \frac{1}{(\tau_1 \cdot \tau_2 \cdot \tau_a)} \quad 5.25$$

$$T_2 = \frac{(\tau_1 + \tau_2 + \tau_a)}{(\tau_1 \cdot \tau_2 \cdot \tau_a)} \quad 5.26$$

$$T_3 = \frac{(\tau_1 \cdot \tau_a + \tau_1 \cdot \tau_2 + \tau_a \cdot \tau_2)}{(\tau_1 \cdot \tau_2 \cdot \tau_a)} \quad 5.27$$

$$T_4 = \frac{k_{sc}}{\tau_2} \quad 5.28$$

$$T_5 = \frac{k_{sc} \cdot \tau_L}{\tau_2} \quad 5.29$$

Equation 5.24 can be expressed in state space as follows for the transfer function:

$$\begin{aligned} \dot{x}_{sc} &= A_{sc} \cdot x_{sc} + B_{sc} \cdot u \\ \dot{\theta}_s &= C_{sc} \cdot x_{sc} + D_{sc} \cdot u \end{aligned} \quad 5.30$$

where the matrices A_{sc}, B_{sc}, C_{sc} și D_{sc} are defined as follow:

$$A_{sc} = \begin{bmatrix} -T_3 & 1 & 0 \\ -T_2 & 0 & 1 \\ -T_1 & 0 & 0 \end{bmatrix} \quad 5.31$$

$$B_{sc} = \begin{bmatrix} T_4 - T_3 \cdot T_5 & 0 \\ -T_2 \cdot T_5 & 0 \\ -T_1 \cdot T_5 & 0 \end{bmatrix} \quad 5.32$$

$$C_{sc} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \quad 5.33$$

$$D_{sc} = \begin{bmatrix} T_5 & 0 \end{bmatrix} \quad 5.34$$

The state space representation of the human vestibular system, comprising the semicircular canal and the otolith organ, is defined as follows:

$$\begin{aligned} \dot{x}_{SV} &= A_{SV} \cdot x_{SV} + B_{SV} \cdot u \\ y_S &= C_{SV} \cdot x_{SV} + D_{SV} \cdot u \end{aligned} \quad 5.35$$

where x_{SV} contains the states of the human vestibular model, and y_S is the detected response. The matrices A_{SV}, B_{SV}, C_{SV} și D_{SV} are represented as follows:

$$A_{SV} = \begin{bmatrix} A_{sc} & 0 \\ 0 & A_{ot} \end{bmatrix} \quad 5.36$$

$$B_{SV} = \begin{bmatrix} B_{sc} \\ B_{ot} \end{bmatrix} \quad 5.37$$

$$C_{SV} = \begin{bmatrix} C_{sc} & 0 \\ 0 & C_{ot} \end{bmatrix} \quad 5.38$$

$$D_{SV} = \begin{bmatrix} D_{sc} \\ D_{ot} \end{bmatrix} \quad 5.39$$

The definitions of vestibular system state error and pilot sensation error are as follows:

$$\begin{aligned} \dot{x}_{err} &= A_{SV} \cdot x_{err} + B_{SV} \cdot u_{sim} - B_{SV} \cdot u_{A/C} \\ err &= C_{SV} \cdot x_{err} + D_{SV} \cdot u_{sim} - D_{SV} \cdot u_{A/C} \end{aligned} \quad 5.40$$

where the simulator and aircraft inputs are u_{sim} respectively $u_{A/C}$.

5.4. Procedure for implementing the motion cueing algorithm

The input signal vector is taken to be:

$$u_{A/C}(s) = \begin{bmatrix} a_{Ax}(s) & \dot{\theta}(s) \end{bmatrix} \quad 5.41$$

By completing the equations and deriving the new terms, a general transfer function is generated. This function links the real motion with the motion of the flight simulator.

$$u_{sim}(s) = W(s) \cdot u_{A/C}(s) \quad 5.42$$

where W is the optimized transfer function matrix that transfers the simulator inputs $u_{sim}(s)$ to the real motion $u_{A/C}(s)$. The transfer function $W(s)$ is defined in matrix form as follows:

$$W(s) = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} \quad 5.43$$

The form of each transfer function W_{ij} is shown in equation (18), in total twelve parameters are used to complete the denominator and numerator respectively. The denominator parameters are unique for each W_{ij}.

$$W_{ij} = \frac{a_5 \cdot s^5 + a_4 \cdot s^4 + a_3 \cdot s^3 + a_2 \cdot s^2 + a_1 \cdot s + a_0}{b_5 \cdot s^5 + b_4 \cdot s^4 + b_3 \cdot s^3 + b_2 \cdot s^2 + b_1 \cdot s + b_0} \quad 5.44$$

Finding the appropriate matrix for the numerical solution is not straightforward and can lead to failure, so the genetic algorithm was used to identify the optimal parameters.

The objective function for this problem is shown in the equation below as follows:

$$F_c = \int_0^{t_f} \left[|z_{sim} - z_0| + |p_{A/C} - p_{sim}| \right] dt \quad 5.45$$

The lift position of the flight simulator is denoted by z_{sim} and the base value, the physical constraint of the flight simulator, is denoted by z_0. And $p_{A/C}$ is a function of the sensed perception (human perception) before and after applying the wash filter. The notation A/C and sim denote the actual motion and the motion of the simulator, respectively. The last time in the equation is defined by t_f.

Figure 5.8 shows an overview of the implemented model. The diagram shows the procedure and the general steps of implementing the optimal constraint filter. Using the basic elements of the classical constraint filter and the transfer functions (equation 5.44), the basis of this filter is prepared, then the cost function (equation 5.45) is considered to reduce the perceptual error and free the flight simulator from its physical constraints, and then the optimisation algorithm is used to obtain the coefficients of the filter functions.

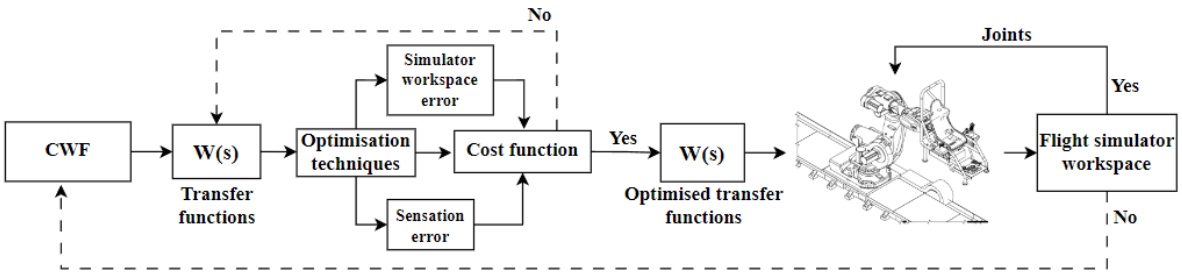


Figure 5.8 Method of implementing the OWF

As mentioned above, the purpose of the OWF based on LQR is to determine a transfer function, $W(s)$, that relates the input given by the simulator motion, u_{sim}, to the input given by the aircraft motion, $u_{A/C}$ according to equation 5.42. The control inputs, linear accelerations and angular velocities, are applied to generate the desired basic motion commands. The optimal washout filter based on LQR method determines the simulator acceleration by minimising the human perception error between the simulator and the real aircraft as well as the linear and angular motion of the platform. The aim of the method is to limit the human perception error and platform motion within the workspace constraints of the motion platform - a serial robotic system.

5.5. Genetic algorithm

Evolutionary algorithms (EA) are robust methods for finding near-optimal solutions, with the ability to handle ill-defined evaluations that have certain properties such as: coupling, time variation, discontinuity, noise and probability [57]. Genetic algorithm (GA) is one of the efficient evolutionary algorithms, which is a heuristic, robust and reliable search method that generates a solution through a process that mimics natural selection and evolution. GA is a stochastic population-based algorithm, and the algorithm's potential to solve complicated problems is an important advantage over classical optimisation techniques.

Each chromosome in the GA population randomly searches the solution space independently and simultaneously in different directions. This makes it ideal for parallel implementation [58]. GA uses genetic operations (selection, crossover and mutation) to evolve solutions. In GA, a population is a set of individual chromosomes [59]. The process of evolution by creating a new population is repeated until a predefined stopping criterion is met. Thus, the algorithm is used to improve the solution provided by the optimal washout filter algorithm based on LQR. The aim is to improve human perception and signal shape tracking while reducing false motion cues. The method takes into account several factors such as simulator limitations, shape tracking criterion, signal accuracy and human threshold in tilt coordination. Figure 5.9 illustrates the scheme of the GA operators used to optimise the motion algorithm, which will be explained next.

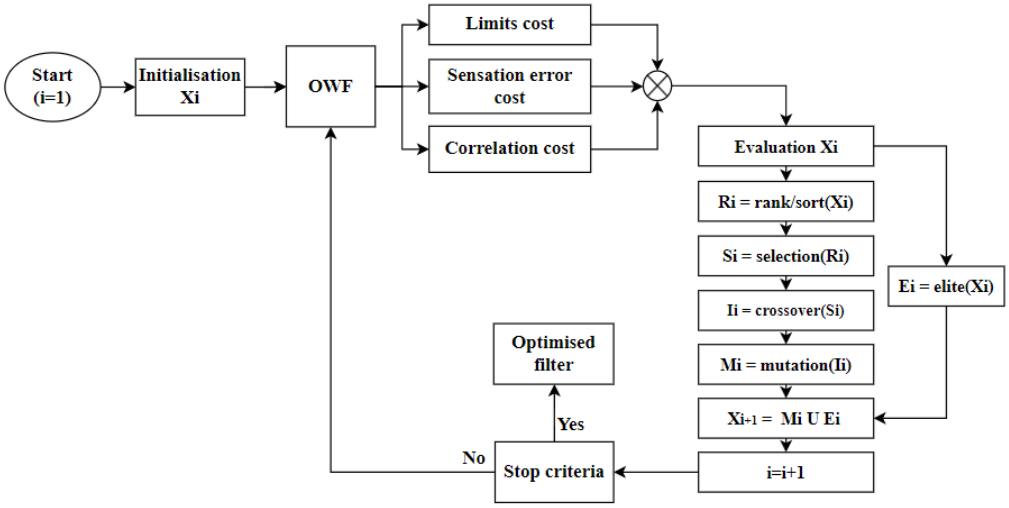


Figure 5.9 Structure of the genetic algorithm

Each solution in its generation is represented by a chromosome made up of genes:

$$A_c^i = \left[A_{c,1}^i, A_{c,2}^i, \dots, A_{c,n}^i\right] \quad 5.46$$

where c is the number of chromosomes in the population and n is the number of genes in each chromosome. To evaluate each individual, it is necessary to establish a defined performance index. The objective function is formulated to rank the performance of each individual. For the proposed method, the overall objective function consists of five sub-problems as follows:

I. Minimisation of translational and rotational errors of human perception between what pilots feel in the real aircraft and in the simulator.

II. Minimise the angular and linear displacements of the motion platform to operate within its physical limits.

III. Minimise the variation of perception errors.

IV. Maximising the correlation coefficient, defined as the Pearson moment correlation coefficient, to increase the traceability of the reference signal shape in the produced signal.

V. Minimise the acceleration and velocity of the moving platform due to physical constraints.

Therefore, the objective function at each i iteration for the c individual is defined as the sum of the sub-objective functions:

$$O(A_c^i) = O_{c\_sener}^i + O_{c\_cor}^i + O_{c\_lim}^i \quad 5.47$$

The chromosomes in the GA are initialised with the individuals obtained from the results of the LQR-based optimal constraint filter algorithm.

The linear and rotational human perception errors between the real pilot and the simulator pilot are defined as follows:

$$er_a(t) = a_{A/C}(t) - a_{sim}(t) \quad 5.48$$

$$er_{\dot{\theta}}(t) = \dot{\theta}_{A/C}(t) - \dot{\theta}_{sim}(t) \quad 5.49$$

To define the human sensation error between real and simulated sensations, a performance index of the integral of the squared error is used in the objective function. The associated objective functions are defined as follows

$$O_{c\_liner}^{i} = k_{a\_er} \int_{t_i}^{t_f} er_a^2(t)dt \quad 5.50$$

$$O_{c\_roter}^{i} = k_{\dot{\theta}\_er} \int_{t_i}^{t_f} er_{\dot{\theta}}^2(t)dt \quad 5.51$$

where k_{a_er} și $k_{\dot{\theta}\_er}$ are the weights that determine the influence of translation and rotational sensation errors in the cost function, and t_i represents the start time and t_f the test completion time. These weights are chosen according to previous research on motion algorithm models [56]. The objective function for the total human sensation error is the sum of the translation and rotation objective functions:

$$O_{c\_sener}^{i} = O_{c\_liner}^{i} + O_{c\_roter}^{i} \quad 5.52$$

The following section defines the objective function of the cross-correlation coefficient, which indicates the dependence of two signals. To accurately follow the true feeling signal, an increased correlation coefficient is necessary. The Pearson correlation coefficient between two signals, y_1(t) and y_2(t) is defined as follows:

$$C_c(y_1, y_2) = \frac{C_v(y_1, y_2)}{\sqrt{C_v(y_1, y_1) \cdot C_v(y_2, y_2)}} \quad 5.53$$

But it can be expressed in another form like this:

$$C_c(y_1, y_2) = \frac{C_v(y_1, y_2)}{\sigma_{y_1} \cdot \sigma_{y_2}} \quad 5.54$$

The covariance of two signals is:

$$C_v(y_1, y_2) = E[(y_1 - E[y_1])(y_2 - E[y_2])] \quad 5.55$$

$$\sigma_{y_i} = \sqrt{E[(y_i - E[y_i])^2]} \quad 5.56$$

where $E[y_i]$ represents the mean of the signals y_i, σ_{y_i} represents the standard deviation of the signal and $E[...]$ representd the desired value

The correlation coefficient between $y_1(t)$ and $y_2(t)$ is:

$$C_c(y_1, y_2) = \frac{E[(y_1 - E[y_1])(y_2 - [y_2])]}{\sigma_{y_1} \cdot \sigma_{y_2}} \quad 5.57$$

Thus, the cross correlation objective function can be defined to provide a solution with a better approximation of the shape of the reference sensation signal thus:

$$O_{c\_lincor}^{i} = k_{a\_cor}\left|1 - C_c(a_{A/C}, a_{sim})\right| \quad 5.58$$

$$O_{c\_rotcor}^{i} = k_{\dot{\theta}\_cor}\left|1 - C_c(\dot{\theta}_{A/C}, \dot{\theta}_{sim})\right| \quad 5.59$$

where k_{a_cor}și $k_{\dot{\theta}\_cor}$ are the linear and angular weights of the correlation coefficient.

The total objective function of the correlation coefficient is defined as follows:

$$O_{c\_cor}^{i} = O_{c\_lincor}^{i} + O_{c\_rotcor}^{i} \quad 5.60$$

As far as the limits of the moving platform are concerned, linear and angular displacements should be kept to a minimum to avoid reaching the limit of the working space. To prevent

extreme linear velocities/accelerations and angular velocities being reached within the motion platform, the limiting objective function must include velocity and acceleration factors.

The limiting objective function is represented as follows:

$$O^i_{c\_linlim} = k_x \int_{t_i}^{t_f} x^2(t)dt + k_v \int_{t_i}^{t_f} v^2_{sim}(t)dt + k_a \int_{t_i}^{t_f} a^2_{sim}(t)dt \quad 5.61$$

$$O^i_{c\_rotlim} = k_\theta \int_{t_i}^{t_f} \theta^2(t)dt + k_{\dot\theta} \int_{t_i}^{t_f} \dot\theta_{sim}{}^2(t)dt \quad 5.62$$

where k_x is a weight for linear displacement limitation, and k_v and k_a are for linear velocity and acceleration limitations. In equation 5.62, k_θ is a weight for angular displacement.

$$O^i_{c\_lim} = O^i_{c\_linlim} + O^i_{c\_rotlim} \quad 5.63$$

These constraints have been incorporated into the simulation block in the form of constraints to restrict movements within the specific limits of the simulator.

The AG selection process is utilised to choose individuals for breeding, giving a higher probability to the most suitable candidates. For each objective function evaluation, a MATLAB/Simulink model is simulated and run in a closed loop over the AG generation, which aims to minimise the objective function. The raw values are scaled by the selection operation using rank scaling as follows:

$$Sc(f_i) = \frac{1}{\sqrt{R_i}} \quad 5.64$$

where R is the rank of individuals in the population, starting from 1 for the best individual. To calculate the objective function, Simpson's rule is applied. In AG breeding, the selection scheme proportional to fitness is adopted. Thus, a chromosome with a higher objective value has a higher probability of being copied in the next generation. Figure 5.10 illustrates the simulation scheme used to evaluate the performance of the optimal constraint filter algorithm. The algorithm is adjusted based on GA and considers several factors in the objective function, as mentioned earlier in this subchapter. The GA operations, including selection, crossover, and mutation (as shown in Figure 5.9), and the transfer functions of the optimal washout filter are simulated to minimize the total objective function.

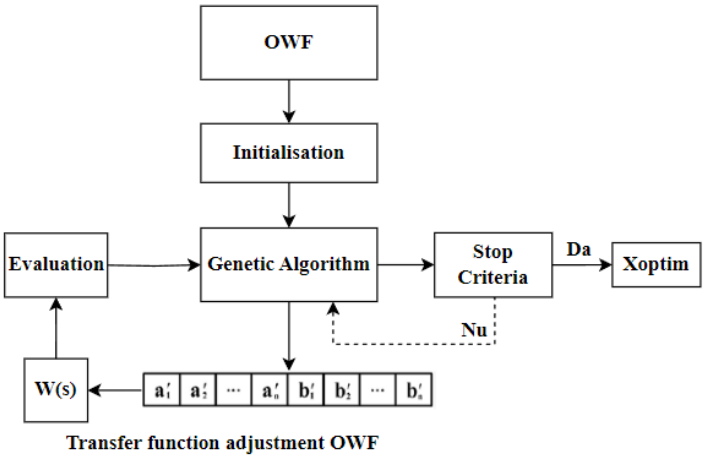


Figure 5.10 Structure of the genetic algorithm for adjusting the transfer functions of the OWF

Designing a closed-loop transfer function by selecting the numerator and denominator coefficients is a simplified process with GA optimization. The optimal washout filter is then optimized by incorporating new aspects and considering their effects. GA is used to adjust the magnitude and phase information in the transfer functions of the optimal washout filter and to improve the zeros and poles in the denominator and numerator. This is useful because each criterion can affect each parameter of the optimal washout filter transfer functions. To modify the poles and zeros of a transfer function, manipulate the numerator and denominator coefficients as follows:

$$W(s) = \frac{B(s)}{A(s)} = \frac{a_0 + a_1 s^{-1} + \cdots + a_m s^{-m}}{b_0 + b_1 s^{-1} + \cdots + b_n s^{-n}} \quad 5.65$$

where $s = j\omega$ is the frequency domain, m is the order of the numerator, n is the order of the denominator and a_i și b_i are the coefficients of the numerator and denominator. Their fit can be represented as follows:

$$W^*(s) = \frac{B^*(s)}{A^*(s)} = \frac{a_0 a'_0 + a_1 a'_1 s^{-1} + \cdots + a_m a'_m s^{-m}}{b_0 b'_0 + b_1 b'_1 s^{-1} + \cdots + b_n b'_n s^{-n}} \quad 5.66$$

where a'_i și b'_i are coefficients for manipulating the primary coefficients of the numerator and denominator, and $a'_0 = b'_0 = 1$ for steady-state error elimination. Then the parameterized transfer function is evaluated by means of the objective function to be optimized by adjusting the coefficients.

Each solution can be represented as a vector

$$X = [a'_1, a'_2, \dots, a'_n, b'_1, b'_2, \dots, b'_m] \quad 5.67$$

where X represents the chromosome, and a_i and b_i are related genes.

The aim of GA optimisation is to minimize the objective function in equation 5.47. The shifting of the zeros and poles considers the correlation coefficient, perceptual error limitations, and nonlinearities that are not accounted for in the LQR method. These additional aspects are included in the objective function to obtain an improved solution for the existing optimal washout filters. Figure 5.10 shows a schematic diagram of the GA fit evaluation used to develop the OWF based on GA optimisation.

The GA considers the optimal washout filter solution as the initial individuals to be evaluated.

$$(X_0^{int})_{1\times(m+n)} = X_n = [1,1,\dots,1] \quad 5.68$$

By changing the numerator and denominator, the zeros and poles move from their original values.

The rate of change for all factors is calculated as follows:

$$\begin{gathered} r \le a'_i \le \frac{1}{r} \\ r \le b'_i \le \frac{1}{r} \\ rX_n \le X_i^t \le X_n \end{gathered} \quad 5.69$$

where r is an arbitrary rate less than 1, but which must not be very far from this value. The selection of r depends on the case under study, and in the present study, after several attempts, the rate was set to $0{,}81$. This setting guarantees that the new poles and zeros remain very close

to the previous solutions in the optimal washout filter algorithm, even in the case of the worst-case solution.

Equation 5.69 defines the parameter ranges for the GA search process used to solve the optimization problem. The simulation process evaluates each parameter set obtained for the washout filter. The GA evolution loop continues the search until the stopping criterion, which is a threshold for the objective function value, is met.

6. Results

This chapter highlights specific results for the optimal washout filter algorithm. The algorithm was formulated, designed, developed, implemented, tested, and validated to improve the flight simulation environment using the IRB 7600-500 serial robotic system-based flight simulator. To validate the applicability of the optimal washout filter and compare it with the modified filter based on information provided by the genetic algorithm, six flights were performed using a dynamic model for a fast-evolving aircraft. In the previous chapter, was implemented the optimal washout filter based on LQR with the centre of rotation of the simulator platform located in the subject's (pilot's) head area to eliminate potential false motion cues. The inputs to the washout filter are the linear acceleration and angular velocity. The vestibular system model, including the semicircular canal and the otolith organ, is part of the filter scheme to eliminate false motion cues. The parameters for the vestibular system, as defined in Chapter 5 for the otolith system and semicircular canal, are shown in the tables below, according to [56].

Table 6.1 shows the specific force parameters for the otolith system and the frequency response of the specific force sensation transfer function for the otolith system is shown in Figure 6.1.

Table 6.1 Specific force parameters for the otolith system

	X	**Y**	**Z**
$\boldsymbol{k_{ot}}$	0.4	0.4	0.4
$\boldsymbol{\tau_1}$ $[s]$	5	5	5
$\boldsymbol{\tau_2}$ $[s]$	0.016	0.016	0.016
$\boldsymbol{\tau_L}$ $[s]$	10	10	10
$\boldsymbol{A_0}$ $[s^{-1}]$	$1/\tau_L$	$1/\tau_L$	$1/\tau_L$
$\boldsymbol{B_0}$ $[s^{-1}]$	$1/\tau_1$	$1/\tau_1$	$1/\tau_1$
$\boldsymbol{B_1}$ $[s^{-1}]$	$1/\tau_2$	$1/\tau_2$	$1/\tau_2$
$\boldsymbol{k_O}$	$k_{ot}\tau_1\tau_2/\tau_L$	$k_{ot}\tau_1\tau_2/\tau_L$	$k_{ot}\tau_1\tau_2/\tau_L$
Prag $[m/s^2]$	0.17	0.17	0.28

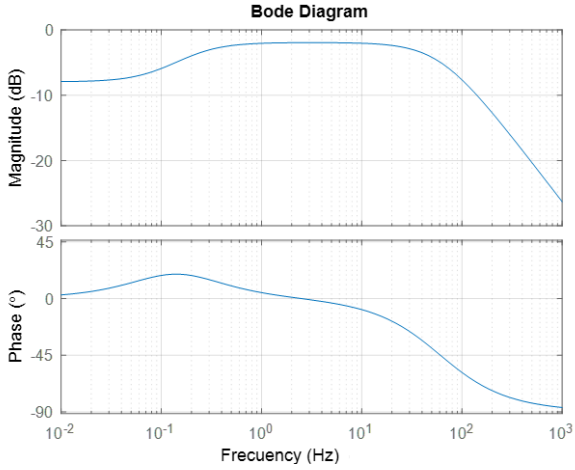


Figure 6.1 Frequency response for the otolith model

Table 6.2 shows the specific parameters of the rotational motion model for the semicircular system, and the frequency response of the rotational sensation transfer function for the semicircular channel is shown in Figure 6.2.

Table 6.2 Specific parameters for rotational motion for semicircular system

	Roll (x)	**Pitch (y)**	**Yaw (z)**
k_{sc}	28.6479	28.6479	35.8099
τ_1 $[s]$	5.73	5.73	5.73
τ_2 $[s]$	0.005	0.005	0.005
τ_L $[s]$	0.06	0.06	0.06
τ_a $[s]$	80	80	80
Threshold $[grad/s]$	2.0	2.0	1.6

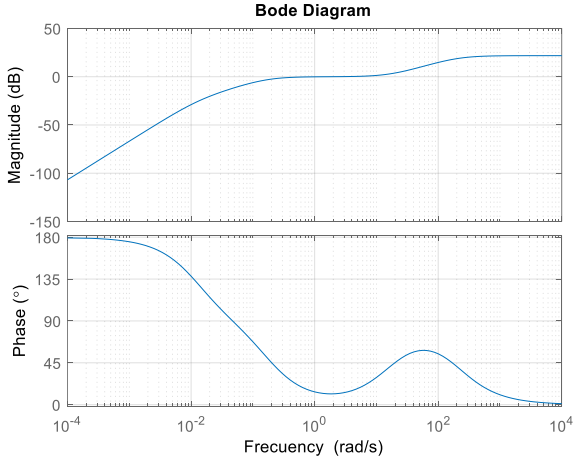


Figure 6.2 Frequency response for the semicircular system

The use of GA presents a significant challenge in selecting appropriate parameters, including population size, mutation rate, and crossover rate. These parameters determine solution accuracy and algorithm convergence speed. Therefore, setting proper crossover and mutation rates is crucial for successful results. The algorithm tests scenarios by achieving optimal crossover or mutation rates through trial and error. A population size of 60 chromosomes is considered optimal for accurate results in the presented experiments. Increasing the population size beyond 60 chromosomes does not noticeably improve the solution. However, the computation time is increased and the convergence speed is

considerably decreased. Additionally, the optimal solution is highly dependent on the initialization of the chromosomes. Therefore, the initial values for each chromosome are selected based on the previous solutions provided by optimal washout filter based on LQR. Evolution is then started using these parameters. It is important to note that this approach aims to maintain consistency with the previous washout filter. The crossover rate of 0.78, mutation rate of 0.25, and 5% of the population size are used for elite breeding and rank scaling, based on research in the literature.

Figure 6.3 shows the evaluation scheme for tuning the genetic algorithm used to develop an optimal washout filter in a MATLAB/Simulink simulation environment.

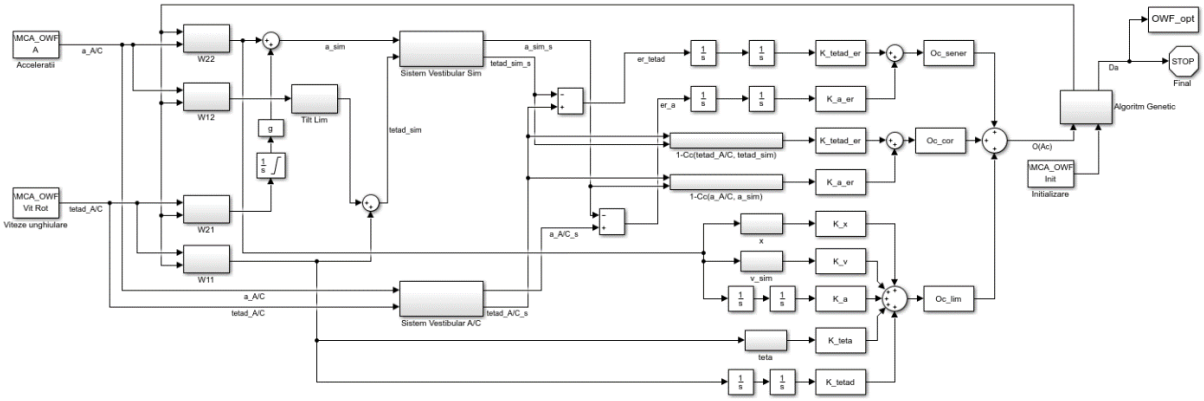


Figure 6.4 Scheme of the GA fit evaluation for the optimal washout filter algorithm

The following section presents the results obtained from the optimal washout filter based on LQR and the modified optimal washout filter based on genetic algorithm. Graphical representations are used to plot flight information and to compare the results of the two filters for linear acceleration, roll, and pitch motion during different flight phases. The results will be presented for short flight sequences of a maximum of 50 seconds, specifically during the cruise and landing phases. The scenario for the six flights comprised one lap of runway 08R LROP, which took approximately 8 minutes per flight. The recorded data were updated and stored 60 times per second in a database. To highlight the shape of the signal recorded by the two filters, a graphical representation was chosen for flights 1 and 2.

Figure 6.5 displays the parameters that describe the flight segment, including latitude, longitude, altitude, heading, and Mach number, during the cruise flight phase sequence for the first flight scenario. The gravitational force was recorded with minimum and maximum values of $g_{min} = 0.3914$ and $g_{max} = 2.2256$.

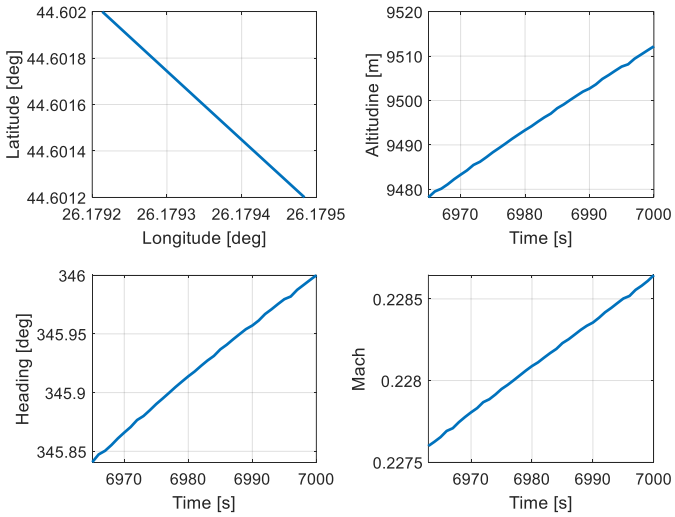


Figure 6.5 Flight scenario (1) - Flight parameters - cruise sequence (1C)

In Figure 6.6, Figure 6.7 and Figure 6.8 it can be seen that the signal recorded by the modified optimal washout filter algorithm based on GA yields a signal shape correction for the states recorded by the original algorithm with a noticeable improvement for the roll motion.

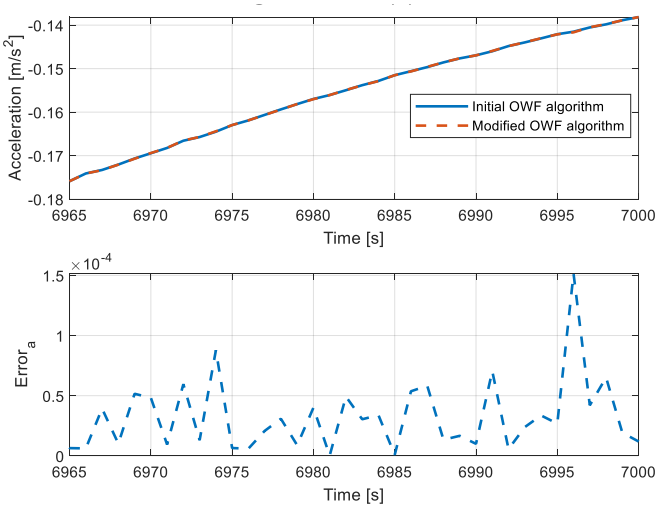


Figure 6.6 Acceleration - comparison between initial optimal washout filter and modified optimal washout filter (GA) (Flight Scenario 1C)

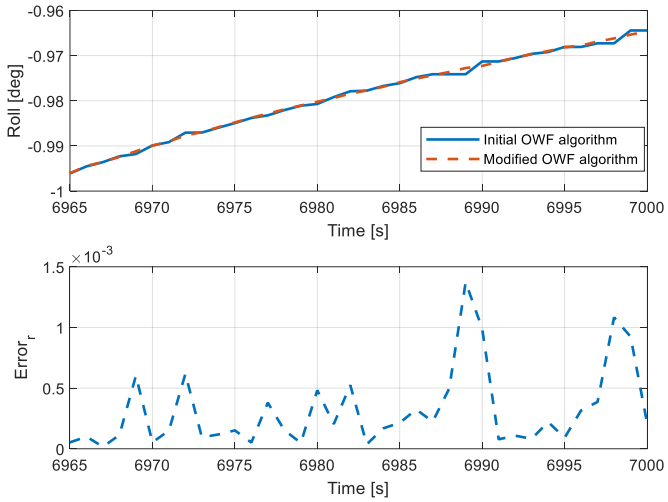


Figure 6.7 Roll angle - comparison between initial optimal washout filter and modified optimal washout filter (GA) (Flight Scenario 1C)

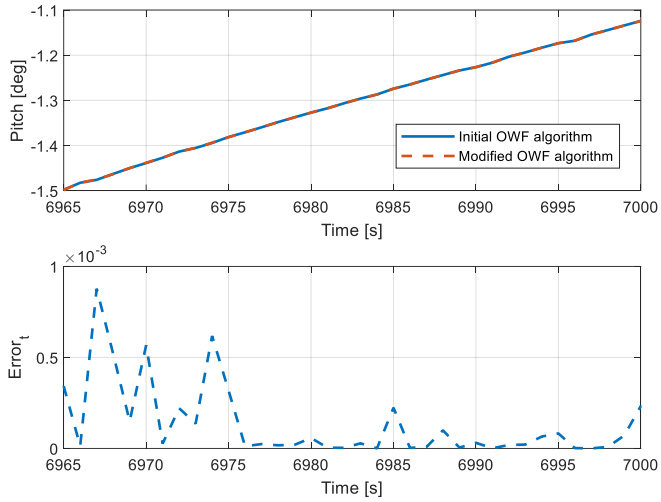


Figure 6.8 Pitch angle – comparison between initial optimal washout filter and modified optimal washout filter (GA) (Flight Scenario 1C)

Figure 6.9 shows the parameters describing the landing flight segment (latitude - longitude, altitude, heading, Mach number) for the sequence chosen for the second scenario, where the gravitational force recorded a minimum value of $g_{min} = 0.5385$ nd a maximum value of $g_{max} = 1.5923$ during the flight.

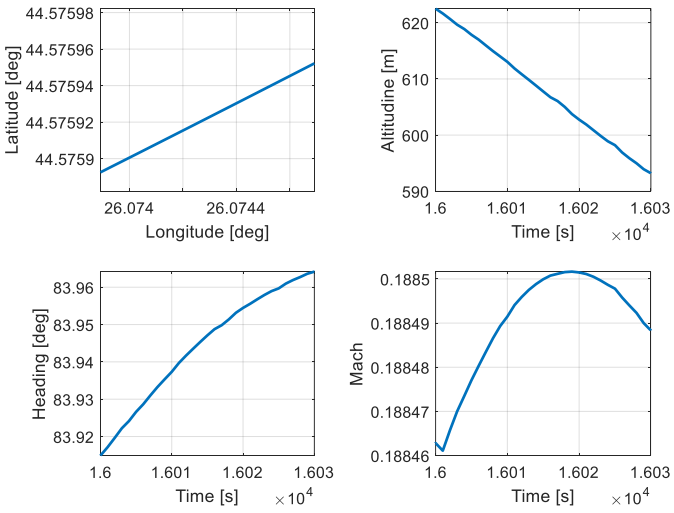


Figure 6.9 Flight scenario (2) - Flight parameters - cruise sequence (2A)

Figure 6.10, Figure 6.11 and Figure 6.12 show the outputs of the modified optimal washout filter compared to the original OWF motion algorithm with the average error between the two algorithms being highlighted as being of order 10^{-4}.

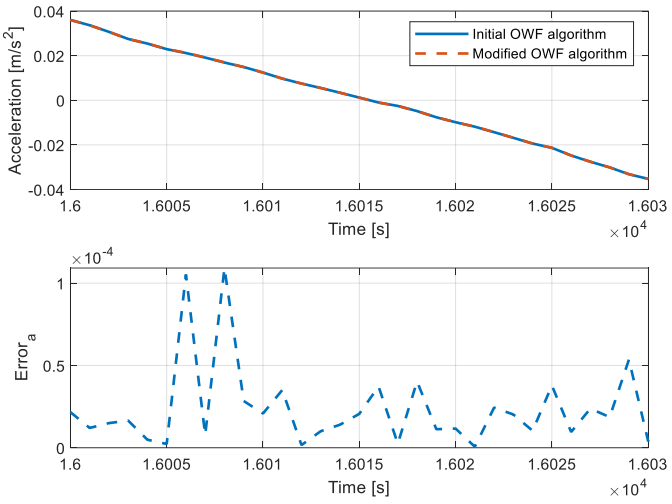


Figure 6.10 Acceleration - comparison between initial optimal washout filter and modified optimal washout filter (GA) (Flight Scenario 2A)

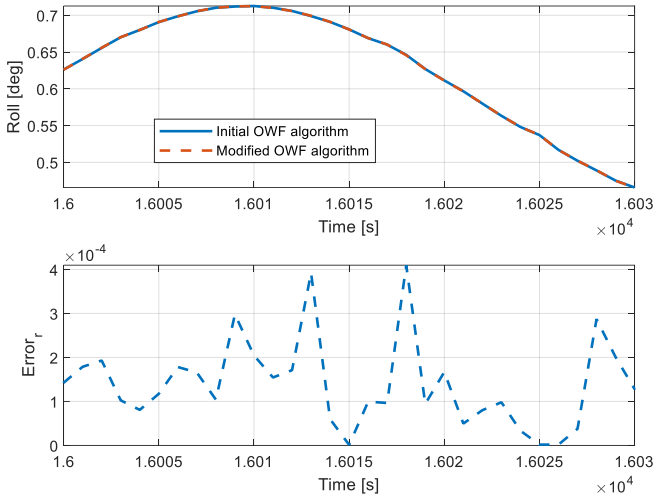


Figure 6.11 Roll angle - comparison between initial optimal washout filter and modified optimal washout filter (GA) (Flight Scenario 2A)

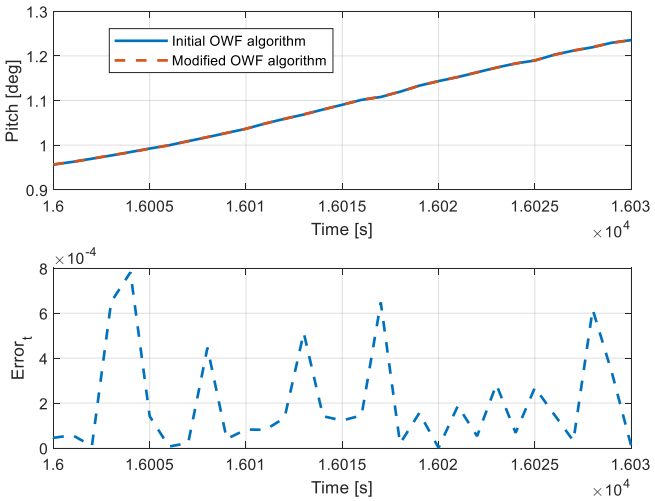


Figure 6.12 Pitch angle – comparison between initial optimal washout filter and modified optimal washout filter (GA) (Flight Scenario 2A)

The results show the effect on the fidelity of the motion indices. From the graphical representations it is observed that for pitch and roll motion the models show almost identical continuity of signal shape, the output signal is able to track the modified wash filter reference signal with a significantly higher degree of correlation, the average error recorded is in the order of $\mathbf{10^{-4}}$.

The modified optimal wash filter algorithm generates better motion with a more realistic feel in a more efficient way by reducing platform displacement. As a result, the simulator can efficiently use the extra space for other possible motions.

7. Conclusions and further developments

7.1. General conclusions

The first part of the PhD thesis presents a study on motion cueing algorithms for flight simulators, based on existing research. A comparative analysis of four types of washout filters was conducted, namely classical, adaptive, optimal control, and robust optimal control, using the results presented in the literature. The advantages and disadvantages of different algorithm designs have been presented to aid in selecting the most feasible option for the flight simulator of the serial robot motion platform. The study determined that the optimal washout filter was selected because it incorporates the vestibular system model, allowing for analysis of human perception error. Additionally, the algorithm tuning is easily accessible through adjustment of the cost function weights, providing a favourable framework for the optimization procedure.

The dynamic simulation system for pilot-in-the-loop applications was mathematically modelled and used as a test platform for the two configurations of the designed and implemented optimal constraint filter. The motion cueing algorithm for the flight simulator was implemented and tested on the ABB IRB 7600-500 robotic system as the motion platform. The article presents the functional structure of the dynamic simulation system and the interaction mode between its components. It defines and plots the reference systems for the simulator platform.

To test the motion algorithm, it was necessary to perform and simulate the inverse kinematics of the serial robot motion platform. The platform is represented as module 4 according to the simulation structure shown in Figure 3.2. The robot's motion was controlled in different directions using various combinations of joint angles. The concept of direct and inverse kinematics was used to determine the end-effector position for fixed joint angles and joint angles for the fixed end-effector. The Broyden-Fletcher-Goldfarb-Shanno (BFGS) optimization algorithm was used to solve the inverse kinematics problem in two simulation environments: MATLAB R2022b and Robot Studio. The simulation results are presented in subsection 3.4.2, and the model's effectiveness can be confirmed by moving along a predefined trajectory of the serial robot end-effector. The discussion also covered the result of the inverse kinematics for various combinations of joint angles, highlighting the differences in the position vector configuration.

The proposed algorithm includes an essential module for scaling and limiting the workspace of the flight simulator. This involves redefining the minimum and maximum limits of the motion platform joints to ensure the simulator remains within safe limitations and constraints during use and testing. Modifications were made to the joint angles of the robot for hardware limitations, specifically for joint q_3, q_4 și q_5, which now have limits of [-180 +40], [-175 +175], and [-90 +90] respectively.

The motion algorithm transforms the acceleration and angular velocity of a simulated vehicle to reproduce high-fidelity motions within the physical limits of the simulator. In the PhD work, the optimal washout filter was designed based on the LQR method, which considers the vestibular system and the motion of the simulator to efficiently reduce the human perception error between the simulator and the real aircraft.

A limitation was applied to the ABB IRB 7600-500 serial robotic system for the translational and rotational channel due to physical constraints. The state vector signals ($u_{A/C}$) used as input to the algorithm model, a scaling was applied to change the signal amplitude uniformly across all frequencies. To achieve this, a third-order polynomial was implemented in the general scheme of the optimal washout filter algorithm. The scaling and limiting units are crucial for ensuring that the motions in the flight simulator remain within physical limits. They work together with the optimal washout filter to uniformly reduce the magnitude of translational and rotational motion signals across all frequencies in the algorithm. This helps to mitigate the effects of workspace limitations on reproducing simulator motions and improves the realism of the motion feel.

Sub-chapter 5.3 presents the mathematical modelling of the vestibular system, which is a crucial component of the optimal washout filter motion algorithm.
Simultaneously, the reference system with the centre of rotation at the pilot's head was selected to eliminate false motions and the sensation of cross-coupling between the simulator's rotation and the pilot's head translational motion. Additionally, it was chosen to reduce the displacement in the workspace of the motion platform.

A new strategy based on evolutionary algorithms has been proposed to improve the optimal washout filter based on LQR. The main objective is to regenerate a signal that can closely track the reference signal, avoid false motion cues, and improve its shape. The genetic algorithm cost function considered several criteria, including the error between the motion sensation recorded by the real pilot and the simulated sensation, the imposition of dynamic simulator constraints, the setting of the human threshold limiter in the pitch coordination, the derivative of the human sensation error, and the cross-correlation coefficient.

The results indicate that the proposed OWF method based on genetic algorithms can comprehensively handle all these aspects. The optimal motion cueing algorithm based on genetic algorithms was implemented using the MATLAB/Simulink software package.

The results obtained demonstrate the superiority of the proposed optimal washout filter based on genetic algorithms due to its better performance, improved human sensation, increased shape tracking factor, and reduced displacement. The proposed optimal washout filter based on GA achieves a balance between human sensation error and shape tracking, resulting in better motion and a more realistic experience for the pilot in the simulator. Additionally, the filter efficiently utilizes the workspace, allowing for more possible motions to be included. The results showed that the algorithm is reliable, robust and efficient in this application for the ABB IRB 7600-500 serial robot motion platform-based flight simulator.

The proposed algorithm has the advantage of being applicable as a tool for any type of simulator, regardless of physical limitations, or for the same simulator in other physical confinements. This can be achieved without requiring significant effort to tune the specific parameters of the motion indication algorithm.

7.2.Original contributions

The PhD work involved designing, implementing, and testing a motion cueing algorithm, an optimal washout filter based on LQR, and an improved version of the filter based on genetic algorithms. This was the first time such algorithms were used for flight simulators based on ABB IRB 7600-500 serial robot motion platforms. The algorithm described in Chapter 2 has only been tested on simulators based on parallel robot motion platforms (Stewart type platforms).

This study introduces a novel aspect by limiting and scaling the ABB IRB 7600-500 type motion platform for the serial robot system. This involves an application of specific procedure for the flight simulator.

Designing and implementing motion cueing algorithms pose a significant challenge, particularly in allocating joint values to control human perception and achieve realistic motion. To mitigate human perception error and create faithful motion in the flight simulator workspace, the washout filter algorithm should be optimally implemented. The objective function is defined by two criteria: minimizing the workspace of the flight simulator and minimizing the human perception error. Two different maneuvers, roll and pitch, are considered. The results presented in Chapter 6 demonstrate that motion occurs within the simulator workspace and the perception error is minimal. The correlations of human perception before and after the application of the proposed eashout filter are higher than 85% for all studied manoeuvres, indicating that human perception does not change significantly when the washout filter is applied.

In the optimal washout filter algorithm, the cost function used maximizes the correlation coefficient, specifically the Pearson moment correlation coefficient. This is done to increase the traceability of the reference signal shape in the produced signal. It is important to allow the regenerated signal to follow the shape of the signal from the real pilot, in addition to decreasing human sensing error. The cost function of the motion cueing algorithms implemented so far has not taken into account this significant aspect.

The proposed washout filter reduces both the actual acceleration in the simulator workspace and the perception system error. This method accurately produces flight simulator platform motions with higher fidelity and efficiency while adhering to the physical limitations of the simulator.

The OWF motion cueing algorithm has been improved by using the genetic algorithm for six degrees of freedom simulator configuration.

7.3. Future research directions

To enhance the motion cueing algorithm, it is recommended to implement and test other optimization methods that can improve the fidelity of motion in various flight scenarios and operating conditions of the flight simulator platform.

Additionally, further research is required to gain a better understanding of the subjective evaluations of test pilots. On top of this, conducting further experiments to investigate the impact of false cues and errors on motion perception fidelity could enhance the algorithm's applicability.

In the future, it will also be important to optimise the initial position of the flight simulator to improve its workspace utilisation for specific simulation tasks.

As a direction for future research, the proposed method implemented in the MATLAB/Simulink simulation environment as a computational program developed for optimizing motion cueing algorithms for flight simulators can be used to evaluate other types of motion cueing algorithms.

Furthermore, evolutionary computational methods can be employed to design the nonlinear scaling unit. It is worth investigating and designing a nonlinear scaling unit based on prediction intervals. Conducting further tests using a flight simulator to analyze the results provided by the proposed method is planned for future phases of this research.

To improve the usability and efficiency of the computational program for optimizing motion cueing algorithms, it is important to develop a graphical user interface with specific inputs for each module of the algorithm. Additionally, the interface should allow for comparison of results from different motion cueing algorithms to aid in selecting the appropriate algorithm for different test scenarios and flight simulator configurations.

Bibliography (selective)

[1] Brain C J., Clayton T D., Ward N J., „Organizing safety related aircraft simulation," Test and Evaluation International Aerospace Forum (2), London, 25-27 June, 1996.

[2] Allerton D.J., „The impact of flight simulation in aerospace," *The Aeronautical Journal,* vol. 114, nr. 1162, pp. 747 - 756, December 2010.

[3] Olivotto C., „Simulators and Training Systems Somewhere in East - Realities, Demands and the Future," în *3rd International Training Equipment Conference and Exhibition Proceedings*, Luxemburg, April 7-11, 1992.

[4] Olivotto C., „The Training Equipment in a Pilot-School Modernization - Flight Simulation, Technologies, Capabilities and Benefits," în *London Royal Aeronautical Society,* May 17-18 1995.

[5] Chira A., Dumitrescu A., Moisoiu C.S., Tanasa C., „Human Performance Envelope Model Study Using Pilot's Measured Parameters," *INCAS Bulletin,* vol. 12, nr. 4, pp. 49-61, 2020.

[6] Bellmann T., Otter M., Hirzinger G., „The DLR robot motion simulator part I: sesign and setup," în *IEEE International Conference on Robotics and Automation,* Shanghai, 2011.

[7] Bellmann T., Otter M., Hirzinger G., „The DLR robot motion simulator part II: optimization based path-planning," în *IEEE International Conference on Robotics and Automation,* Shanghai, 2011.

[8] Teufel HJ., Nusseck HG., Beykirch KA., Butler JS., Kerger M., Bülthoff HH., „MPI motion simulator: development and analysis of a novel motion simulator," în *AIAA Modeling and Simulation Technologies Conference and Exhibit,* South Carolina, 2007.

[9] Chira A.I., Bălașa R.I., Bîlu M.C., Iordache C., Moisoiu C.S., Munteanu C., „Design and Analysis of a Flight Simulator based on a Robotic Motion Platform," în *8th International Workshop on Numerical Modelling in Aerospace Sciences*, Bucharest, 25 May, 2022.

[10] Chira A.I., Bălașa R.I., Bîlu M.C., Iordache C., „Workspace Optimization of A Flight Simulator based on a Robotic Motion Platform," în *8th International Workshop on Numerical Modelling in Aerospace Sciences*, Bucharest, 25 May, 2022.

[11] Chira A.I., Bîlu M.C., Bălașa R.I., Iordache C., „Enhancing Realism in Robotic Flight Simulators: a case study on the implementation analysis of a washout filter," în *34th DAAAM International Symposium on Intelligent Manufacturing and Automation*, Vienna, October, 2023.

[12] Bogos S., Chira A., Udrea I.P., Sandu V., Draghici T., „Comprehensive Analysis of Aircraft Dynamics Stability with SCSim: Integration and Assessment," *INCAS Bulletin,* vol. 15, nr. 4, pp. 41-51, 2023.

[13] Grant P. R. , Reid L. D. , „Motion washout filter tuning: Rules and requirements," *Journal Aircraft,* vol. 34, nr. 2, pp. 145-151, 1997.

[14] Sivan R., Ish-Shalom J., Huang J.-K., „An optimal control approach to the design of moving flight simulators," *IEEE Transactions on Systems, Man and Cybernetics,* vol. 12, nr. 6, pp. 818-827, 1982.

[15] Parrish R. V., Dieudonne J. E., Bowles R. L., & Martin Jr D. J., „Coordinated adaptive washout for motion simulators," *Journal of Aircraft,* vol. 12, nr. 1, pp. 44-50, 1975.

[16] Asadi H., Mohamed S., Lim C. P., Nahavandi S., „Robust optimal motion cueing algorithm based on the linear quadratic regulator method and a genetic algorithm," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* pp. 1-17, 2016.

[17] Chen S. H., Fu L.-C., „An optimal washout filter design with fuzzy compensation for a motion platform," *18th IFAC World Congress, IFAC Proceedings,* vol. 44, nr. 1, pp. 8433-8438, 2011.

[18] Gharib M. R., „Comparison of robust optimal QFT controller with TFC and MFC controller in a multi-input multioutput system," *Reports in Mechanical Engineering,* vol. 1, nr. 1, pp. 151-161, 2020.

[19] Huang C., Fu L., „Human Vestibular Based (HVB) Senseless maneuver optimal washout filter design for VR-based motion simulator," *IEEE International Conference on Systems, Man and Cybernetics,* pp. 4451-4458, 2006.

[20] Kong X.-T., Zhu Y.-C., Di Y.-Q., Cui H.-H., „Methods to determine optimal washout position for single and multioccupant motion simulator," *Cybernetics and Information Technologies,* vol. 16, nr. 1, pp. 173-187, 2016.

[21] Arioui H., Nehaoua L., Amouri A., „Classic and adaptive washout comparison for a low cost driving simulator," *Proceedings of the 2005 IEEE International Symposium on Mediterrean Conference on Control and Automation,* pp. 586-591, 2005.

[22] Chen S. H., Fu L. C., „An optimal washout filter design for a motion platform with senseless and angular scaling maneuvers," *IFAC Proceedings,* vol. 44, nr. 1, pp. 8433-8438, 2010.

[23] Wang X. L., Li L., Zhang W. H., „Research on fuzzy adaptive washout algorithm of train driving simulator," *Tiedao Xuebao/Journal of the China Railway Society* , vol. 32, nr. 2, pp. 31-36, 2010.

[24] Yang Y., Huang Q.-T., Han J.-W., „Adaptive washout algorithm based on the parallel mechanism motion range," *Xi Tong Gong Cheng Yu Dian Zi Ji Shu/Systems Engineering and Electronics,* vol. 32, nr. 12, pp. 2716-2720, 2010.

[25] Moavenian M., Gharib M. R., Daneshvar A., Alimardani S., „Control of human hand considering uncertainties," *IEEE International Conference on Advanced Mechatronic Systems,* pp. 17-22, 2011.

[26] Kim K. D., Kim M. S., Moon Y. G., Lee M. C., „Application of vehicle driving simulator using new washout algorithm and robust control," *SICE-ICASE International Joint Conference,* pp. 2121-2126, 2006.

[27] Salehi Kolahi M. R., Gharib M. R., Koochi A., „Design of a robust control scheme for path tracking and beyond pull-in stabilization of micro/nano-positioners in the presence of Casimir force and external disturbances," *Archive of Applied Mechanics,* vol. 91, nr. 10, pp. 4191-4204, 2021.

[28] Nehaoua L., Amouri A., Arioui H., „Classic and adaptive washout comparison for a low cost driving simulator," în *IEEE International Symp. Mediterranean Conference Control Automation Intell. Control,* Limassol, 2005.

[29] Nehaoua L., Arioui H., Mohellebi H., Espie S., „Restitution movement for a low cost driving simulator," în *Proceeding American Control Conference,* Minneapolis, 2006.

[30] Nehaoua L., Mohellebi H., Amouri A., Arioui H., Espié S., Kheddar A., „Design and Control of a Small-Clearance Driving Simulator," *IEEE Transactions on Vehicular Technology,* vol. 57, nr. (2), pp. 736-746, 2008.

[31] Arioui H., Hima S., Nehaoua L., Bertin R. J. V., Espié S. , „From design to experiments of a 2-DOF vehicle driving simulator," *IEEE Transactions on Vehicular Technology* , vol. 60, nr. (2), pp. 357-368, 2011.

[32] Reid L.D., Nahon M.A. , „Flight simulation motion-base drive algorithms: Part 1. Developing and testing the equations," Technical Report 296, Toronto, 1985.

[33] Reid L.D., Nahon M.A., „Flight simulation motion-base drive algorithms: Part 2. Selecting the system parameters," Technical Report 307, Toronto, 1986.

[34] Reid L.D., Nahon M.A., „Flight simulation motion-base drive algorithms. Part 3: Pilot evaluations," Technical Report 319, Toronto, 1986.

[35] Reid L. D. , Nahon M. A., „Response of airline pilots to variations in flight simulator motion algorithms," *Journal Aircraft,* vol. 25, nr. 7, pp. 639-646, 1988.

[36] Nahon M. A. , Reid L. D. , „Simulator motion-drive algorithms - A designer's perspective," *Journal of Guidance, Control, and Dynamics,* vol. 13, nr. 2, pp. 356-362, 1989.

[37] Zacharias G. L. , „Motion Cue Models for Pilot-Vehicle Analysis," AMRL-TR-78-2, Ohio, USA, 1978.

[38] Hosman R. J. A. W., J. C. van der Vaart, „Vestibular models and thresholds of motion perception: Results of tests in a flight simulator," Delft, The Netherlands, 1978.

[39] Telban R. J., Wu W., Cardullo F. M., „Motion cueing algorithm development: Initial investigation and redesign of the algorithms," Techical Report NASA/CR-2000-209863, Hampton, VA, USA, 2000.

[40] Nahon M.A., Reid L.D., „Simulator motion-drive algorithms: a designer's perspective," *Journal of Guidance, Control and Dynamics,* vol. 13, nr. 2, pp. 356-362, 1990.

[41] Reid L.D., Nahon M.A., Kirdeikis J. , „Adaptive simulator motion software with supervisory control," *Journal of Guidance, Control and,* vol. 15, nr. 2, pp. 376-383, 1992.

[42] FAA - Federal Aviation Administration, „Flight test guide for certification of transport category airplanes," AC 25-7C, 2012. [Interactiv]. Available: http://www.faa.gov/documentLibrary/media/Advisory_Circular/AC_25-7C.pdf. [Accesat 12 September 2022].

[43] Garrett N.J.I., Best M.C., „Driving simulator motion cueing algorithms – a survey of the state of the art," *10th International Symposium on Advanced Vehicle Control,* 2010.

[44] Chira A.I., Afloare A.I., Apostolescu A., Munteanu C.E., „Inverse kinematic solution of a 6 DoF serial manipulator," în *Proceedings of the 30th DAAAM International Symposium,* Vienna, Austria, 2019.

[45] Denavit J., Hartenberg R. S., „A kinematic notation for lower-pair mechanisms based on matrices," în *Journal of Applied Mechanics,* 1955.

[46] MATLAB Robotics Toolbox, [Interactiv]. Available: https://www.mathworks.com/help/robotics/. [Accesat 14 09 2022].

[47] Corke P., „Robotics, Vision & Control - ROBOTICS Toolbox," [Interactiv]. Available: https://petercorke.com/toolboxes/robotics-toolbox/#. [Accesat 14 09 2022].

[48] ABB Robotics - Product specification IRB 7600, ABB, [Interactiv]. Available: https://new.abb.com/products/robotics/robots/articulated-robots/irb-7600. [Accesat 12 09 2022].

[49] Rubio J.D., Aquino V. , Figueroa M. , „Inverse kinematics of a mobile robot," *Neural Computer & Applications,* vol. 23, nr. 1, pp. 187-194, 2013.

[50] ABB Robotics - Robot Studio, ABB, [Interactiv]. Available: https://new.abb.com/products/robotics/robotstudio. [Accesat 14 09 2022].

[51] ABB - Rapid Overview Manual, [Interactiv]. Available: https://new.abb.com/products/robotics/robotstudio. [Accesat 14 09 2022].

[52] Schreiber G. , „Control for redundant robot systems: User and task-oriented use of redundancy," 2004.

[53] Van Egmond A.A., Groen J. J., Jongkees L. B. W., „The Mechanics of the Semicircular Canal," *Journal of Physiology,* vol. 110, pp. 1-17, 1949.

[54] Howard I.P., „Handbook of Perception and Human Performance, The Vestibular System," 1986.

[55] Meiry J.L., „The Vestibular System and Human Dynamic Space Orientation," Sc.D. Thesis, Cambridge, MA, 1965.

[56] Telban R. J., Cardullo F. M., „Motion cueing algorithm development: Human-centered linear and nonlinear approaches," NASA Langley Research, Techical Report CR-2005-213747, Hampton, VA, USA, 2005.

[57] Fleming P.J., Pyrshouse R.C., „Evolutionary algorithms in control systems engineering: A survey," în *Control Engineering Practice,* 2002.

[58] Yang X.S., „Chapter 5 Genetic Algorithms," în *Nature-Inspired Optimization Algorithms,* 2014.

[59] Johar F.M., „A review of genetic algorithms and parallel genetic algorithms on graphics processing unit (GPU)," în *Proc. IEEE Int. Conf. Control Syst. Comput. Eng. (ICCSCE)*, Penang, 2013.

[60] Schmidt S.F., Conrad B., „Motion drive signals for piloted flight simulators," *CR-1601 NASA Contractor Report,* 1970.

[61] Reid L.D., Nahon M.A., „Flight simulation motion-base drive algorithms part II: selecting the system parameters," *N307, UTIAS Report,* 1986a.

[62] Reid L.D., Nahon M.A., „Flight simulation motion-base drive algorithms part III: pilot evaluations," *N319, UTIAS Report,* 1986b.