MSA UNIVERSITY ENGINEERING JOURNAL

Motion Cueing Algorithm for Flight Simulator: Interfacing and Pilot Controls Implementation

Ahmad Essameldin Ahmad [a], **Omar Ashraf Abuzied** [b],

Amgad M. Bayoumy Aly [c], **and Ahmed Badawy** [d]

October University for Modern Sciences and Arts, MSA, 6th of October, Giza, Egypt
E-mail: [a.] ahmad.essameldin@msa.edu.eg , [b,] omar.ashraf20@msa.edu.eg,
[c.] ambayoumy@msa.edu.eg, [d.] ahbadawy@msa.edu.eg

Abstract

Flight simulation nowadays is essential to the training of pilots in both civil and military fields. Generally, the use of parallel motion mechanisms has become the standard for human-in-the-loop simulation. However, the realization of a flight simulator with reasonable fidelity is only achievable using a well-tuned motion cueing algorithm that conveys accurate, real-time cues to the user while maintaining safe use of the available platform workspace. In this paper, the necessary setup and interfacing is done with the X-Plane simulation environment for the purposes of testing and tuning the parameters of a developed motion cueing algorithm. In addition, the design and implementation of prototype pilot controls, namely the yoke and rudder pedals are also undertaken to fulfill interfacing objectives and are integrated into a fixed-seat simulator platform to conduct a full pilot-in-the-loop simulation. MATLAB/Simulink is then used for interfacing hardware pilot controls and the simulation environment as well as the real-time tuning of motion cueing parameters.

Keywords: Flight Simulator, Human-in-the-Loop, Motion Cueing, Pilot Controls, Interfacing

ENGINEERING JOURNAL Volume 2 Issue 2
Received Date January 2023
Accepted Date March 2023
Published Date March 2023
DOI: 10.21608/MSAENG.2023.291919

1. Introduction

Flight simulators nowadays are not only used by professionals for training in the civil and military fields, but the availability of flight simulation software such as Microsoft's Flight Simulator and X-plane, encouraged many individuals and companies to create renditions of flight simulator to enhance the user experience. The fidelity of "do-it-yourself" flight simulators will never compare to professionally designed, built-to-standard examples, but for the recreational applications mentioned above as well as early flight controls training, the unreasonable cost of high-fidelity machinery can be understandably difficult to justify.

Another issue with the commercial flight simulator that this paper aims to solve is that it's too proprietary. That will be solved by using certified and available public flight simulator software (i.e., Microsoft flight Simulator, X-plane, or Flight gear).

For Human-in-the-loop applications such as driving and flight simulation, motion platforms are used to immerse the user into the simulated experience by giving the illusion of being in the real vehicle. One issue, however, is that the operational workspace of the platform is severely limited compared to the vehicle being simulated, and many of the manoeuvres that can be performed within the simulation environment are difficult or sometimes even impossible to replicate due to these limitations. Therefore, a motion cueing algorithm (MCA) is needed. MCAs are a set of transformations based on which a suitable strategy is implemented that focuses on achieving two main objectives: Providing correct motion cues to the user on the motion platform that feel realistic, and Maintaining fidelity while commanding a set of motions that are realizable and within the workspace of the simulator.

The classical washout filter is perhaps the most commonly employed MCA in commercial driving and flight simulators. This approach is described by [1] as consisting of high and low pass filters connected linearly. The filter parameters such as damping ratios and break frequencies are determined empirically by trial and error.

(Rehmatullah, 2017) states the main limitations regarding the classical washout algorithm being the trial-and-error method required for determining the break frequencies and the damping ratios for each filter which can prove to be difficult to set up for inexperienced simulator integrators [2]. An adaptive washout algorithm was proposed and developed for a flight simulator at the Langley research center to create coordinated motions between the translational and rotational channels of the platform using two nonlinear filters [3]. The vestibular system is added to the algorithm to reduce the number of false cues and to improve the quality of simulation [4].

The issue of sub-optimal usage of the motion platform workspace left more to be desired in the research field of motion cueing algorithms. The optimal motion cueing algorithm was the next to be introduced. First developed by [5] the theory was to base the control of the motion cueing on the human perception and how the sensation on the pilot relates to the desired vestibular system dynamics. The optimal control-based motion cueing algorithm as the natural progression of the adaptive algorithm is defined considering the human perception aspect of the simulation [6].

Model predictive control (MPC) algorithm is used to predict and plan a motion trajectory for the motion platform based on the perceived vehicle acceleration [7]. MPC algorithm was developed further by creating an actuator-constraint-based model where actuator limits were used as hard constraints integrated directly into the model that is implemented into a digital controller and on a six DOF platform [8].

2. Interface Development and Simulation

2.1. System Outline

System integration requires a well-defined and organized structure based on which the architecture is determined. This section reflects the findings of the human-in-the-loop survey onto the proposed system architecture. For a flight simulator, the goal is to immerse the user in the simulation environment. This means giving the pilot full control over the system by essentially having the entire experience revolve around them. Figure 1 shows the system outline for the proposed simulator.

The user interacts with the pilot control peripherals that are designed and implemented to enter the flight commands into the simulation environment, which in this case is X-Plane. The pilot control devices that are implemented as part of the simulator include a yoke and rudder pedals. The feedback loops consist of several components, namely the visual and audio cues, peripherals force feedback, and most importantly the motion cueing system made up of the developed motion cueing algorithm and the motion platform's inverse kinematics both of which will be analysed in coming sections. Another component of the feedback loop is the human perception model, otherwise known as the vestibular system.

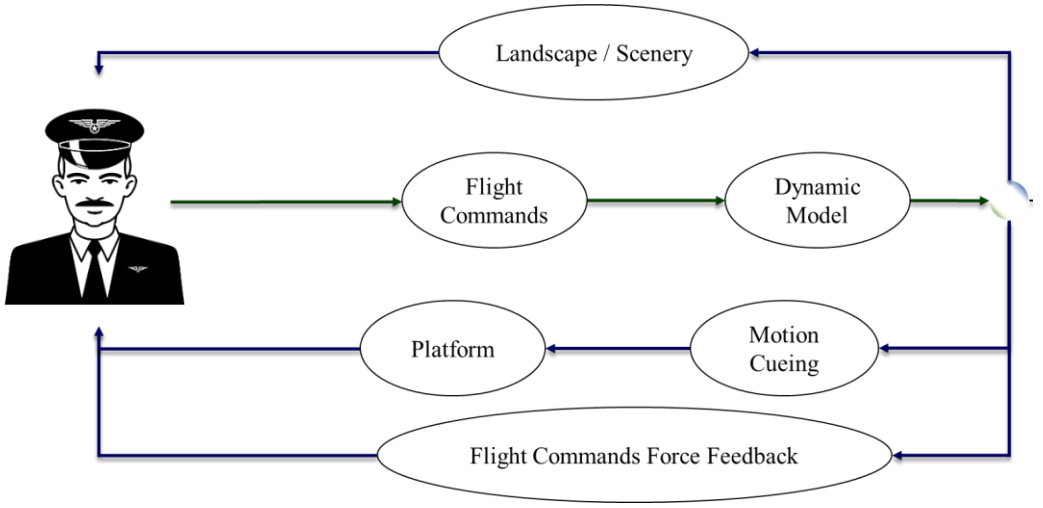


Fig. 1. System Outline

2.2. Interfacing

This paper includes different subsystems responsible for providing the feedback to pilot. These subsystems must be coherent with each other so that no false sensation is

transmitted to the pilot. Thus, interfacing between subsystems coherently and precisely is fundamental. Figure 2 introduces the subsystems in this paper and the communication protocol between them.

Microcontrollers that are responsible for receiving the flight commands from the pilot are connected to MATLAB through serial communication. Flight commands are then sent to the simulation environment (X-plane), which sends the visual and audio cues to the displays and the speakers. X-plane calculates the dynamic model of the aircraft then sends it back to MATLAB. After receiving the dynamic model data, a SIMULINK model is responsible for calculating the required accelerations and velocities. It delivers the same sensation to the pilot (motion cueing). Data are fed to a system that calculates the inverse kinematics of the platform. This communication is done through User Datagram Protocol (UDP) communication. On the other hand, the data extracted from the simulation environment help in creating the haptic feedback to the input peripherals. These data are then fed to the microcontrollers to control the actuators that are responsible for generating the force feedback. After calculating the inverse kinematics for the platform, the required motor angles are fed to the motion platform microcontroller to control the actuators.

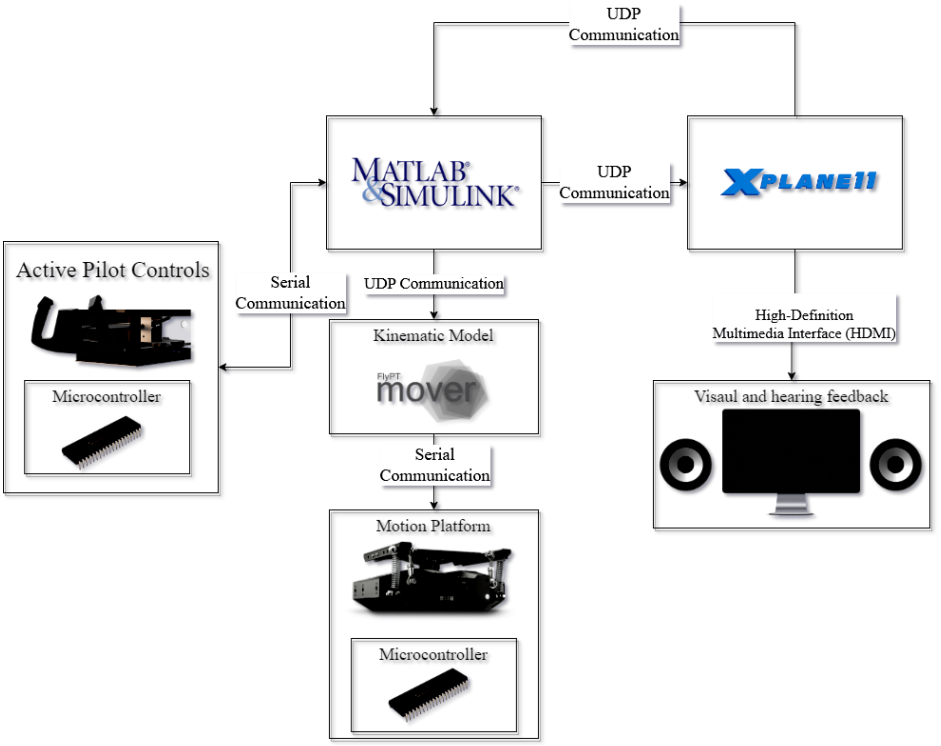


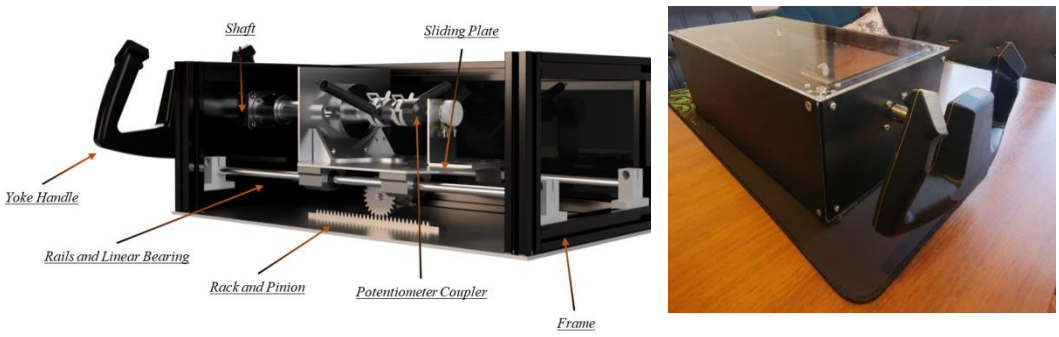
Fig. 2. Subsystems Interfacing

3. Pilot Control Peripherals and Hardware Setup

3.1. Yoke

Roll and pitch are two essential attitudes for any aircraft dynamics. They are controlled by the ailerons and the elevators, respectively based on the pilot's commands. There are several types of control unit to control both ailerons and elevators; however, they all serve the same function.

The yoke inside the Boeing 737 is responsible for controlling the roll and pitch attitudes of the aircraft. It can move towards or away from the pilot to control the elevators (pitch) and rotate 90o to the left and right to control the ailerons (roll). Therefore, the designed yoke should be able to deliver the two degree-of-freedoms smoothly. According to [9] and [10], the pilot exerts maximum force of (Pull: 230N, Push: 210N) on the yoke in the pitch control, and (Left and Right: 20N) in the roll control. Thus, the yoke should stand forces that can reach up to 230N. For the first degree-of-freedom (pitch), the travel range is 18cm, while for the second degree-of freedom (roll) the rotation degree should be 90o clockwise and 90o counter-clockwise. The design of the yoke was implemented to serve the previous specifications, Figure 3.



(a) Internal view (b) External view

Fig. 3 Yoke pilot control

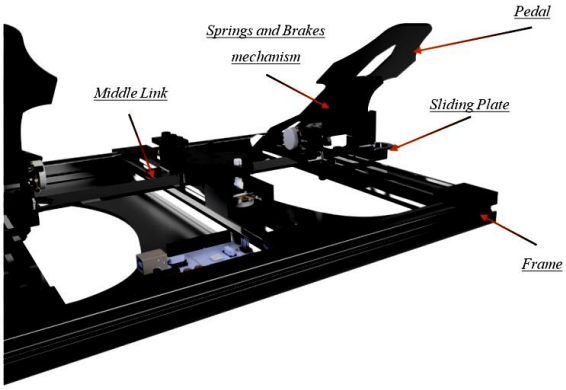
3.2. Rudder Pedals

The third essential attitude parameter is the yaw of the aircraft which is controlled by the rudder pedals. Figure 4 shows the rudder as well as the pedals that control them. In counter to the yoke, rudder pedals come with the almost the same design for all aircrafts. The pedals are not only responsible for controlling the rudder but also controlling the toe brakes, which are responsible for the manoeuvres of the aircraft during taxi.

The nature of the rudders motion is that it rotates to the right and to the left around a fixed axis, to accomplish the control of this motion using pedals. A precise and robust mechanism must be designed. However, first, a set of specifications of the design must be considered:

- When the right pedal goes up the left must go down and vice-versa.
- The pedals should withstand up to a thousand Newton according [9].
- The range of motion should be 15cm upward and 15cm downward.
- The angle of the pedals should be 45^{o}.

The cockpit ergonomics must be considered meaning that the lateral distance between the two pedals should be around 35 cm.



(a) Rudder Pedals Main Parts

(b) Rudder Pedals Final Product

Fig. 4 Rudder Pedals

3.3. Development of Static Flight Simulator Platform

For the purposes of testing and integration of the pilot control peripherals, a static platform is used. This platform is responsible for providing the initial experience for a flight simulator. The platform used in this paper was originally designed and manufactured by [11] and then was developed by [12]. It was responsible for simulating the driving experience equipped with a steering wheel and pedals for automobiles driving control purposes, Fig. 5a. The platform driving peripherals were extracted allowing for the assembly and installation of flight peripherals, Fig.5b.

3.1. Prototype Design of Motion Platform

For the prototype design, it was implemented just to materialize a vision of the design of the final platform as a proof of concept. The prototype design of the platform is based upon a 2-DOF Revolute-Revolute-Spherical (RRS) parallel manipulator. That is, it will have two motor to enable the two rotational motions of pitch and roll, where the motor shaft is considered the first joint and it is active. It is followed by a passive revolute joint, and finally connected to a 3-DOF spherical joint. In addition, a passive universal joint is used to complete the triangle of mounting points in order to constrain the platform in 3D space. The joint specification is shown in table 1. Figures 6 and 7, show the CAD design of the platform, and the manufactured prototype.

(a) Original Vehicle Simulator (b) Flight Peripherals

Fig. 5 Static Flight Simulator Platform

Table 1. Joint Specifications

Joint Type	Degrees of Freedom	Rotational DOF	Translational DOF
Revolute	1	1	0
Spherical	3	3	0
Universal	2	2	0

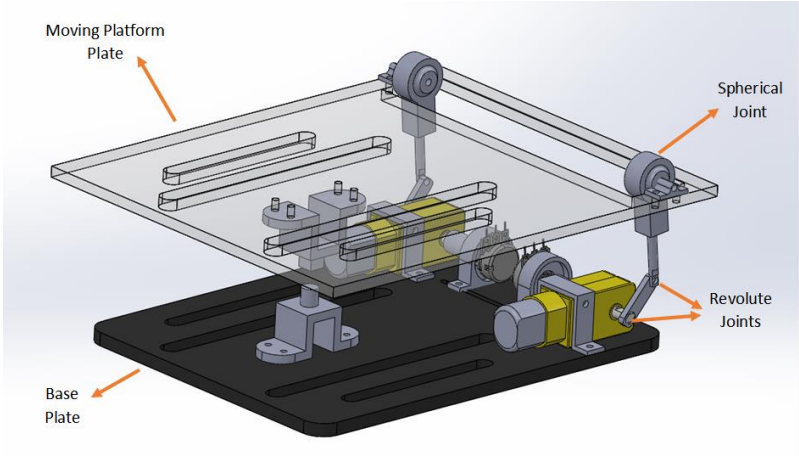


Fig. 6 Platform Prototype CAD Design

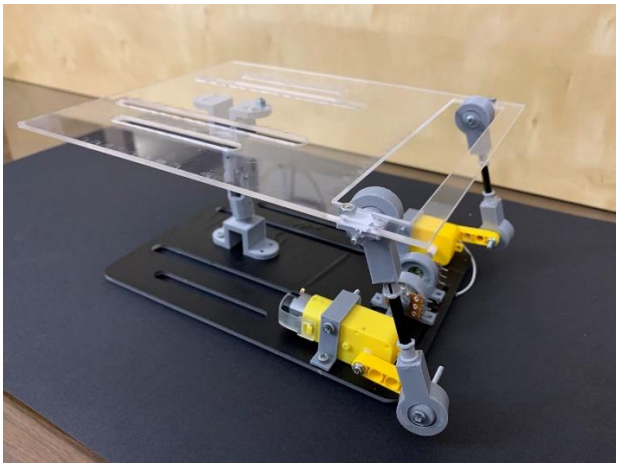


Fig. 7 Motion Platform Prototype Hardware

4. Motion Cueing Algorithm

Figure 8 shows a typical classical washout filter algorithm block diagram. The system inputs are extracted from the simulation environment/program namely the translational accelerations and the angular velocities of the airplane. The filter is divided into three channels: translational, tilt, and rotational channels. A high pass filter is used to extract the fast dynamics, i.e., sudden changes in acceleration; this is extracted from the transient response to ensure that the platform remains within its operational limits. The accelerations are then double integrated to obtain the translational displacement of the platform.

The sustained accelerations which follow the transient phase, and are described by the roll and pitch, are low pass filtered to determine the tilt that the platform needs to maintain. This is directly connected to a limiter referred to as tilt coordination. Finally, the rotational channel contains another high-pass filter which captures sustained motions during transitional phases, and once again, the signal is integrated to produce the angular displacement. A summator adds the results from both the tilt and rotational channels for the final angular displacement magnitude which is outputted to the simulator actuators as length commands.

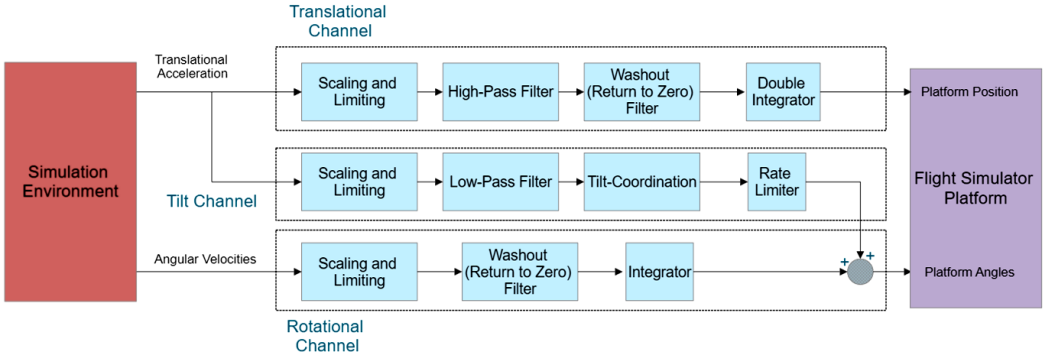


Fig. 8 Classical Washout Filter MCA

5. Results and Discussion

This section showcases the results obtained from the experimental testing of the system interfacing and the real-time deployment of the motion cueing algorithm.

5.1. Interfacing Analysis

Testing the interfacing setup discussed in section 2 along with the hardware setup discussed in section 3, is essential to verify the system validity. Therefore, a specific scenario must be created to analyse the results that come out of it and thus evaluate the system tolerances.

5.1.1. Experimental Setup

To find the possible errors that may occur during the interfacing process, a specific scenario is required. This scenario tests the ailerons, elevators, and rudder response and their effect on the aircraft attitude. This scenario is controlled by the pilot without any autopilot intervention. Table 2 shows the scenario characteristics.

Table 2. Interfacing Experiment: Flight Scenario Characteristics

Aircraft Model	**Boeing 737**
Location	Cairo international Airport
Weather conditions	Clear /wind speed: 0.0 kts

The trajectory of the aircraft must fulfil all monitored variables. Therefore, a departure scenario that includes manoeuvres of the aircraft during taxi and take-off to test the rudder pedals, lifting the aircraft's nose to test the yoke control on the elevators, and roll around its axis to test the yoke's control on the ailerons as shown figure 9.

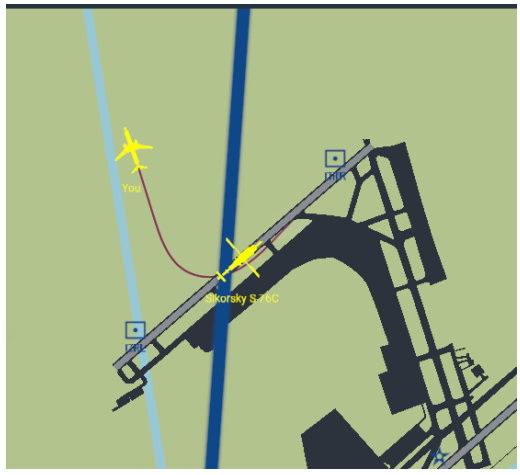


Fig. 9 Interfacing Experiment: Flight Scenario Trajectory

5.1.2 Results

During take-off on the runway, the rudder pedals are the main concern at this time of the scenario. They are used to control the heading of the aircraft. Figure 10 shows the variation between the data sent from the pedals to MATLAB and the received data from X-plane that's being sent back to MATLAB which compares between the two signals to find the errors and tolerances. The effect of the pedals appears in the first 24 seconds where the aircraft at the runway just before take-off. Figure 10 shows the two signals at the same graph along with a green region that indicates the tolerance.

Figure 11 shows the minor difference between the readings with a tolerance reaches 15% at the worst case. The second control parameter is the yoke's pitch level; the pitch takes action when the aircraft reaches to a specific speed when the pilot pulls the yoke to raise the aircraft's nose. Figure 12 shows the start of the pitch effect from the 24th second.

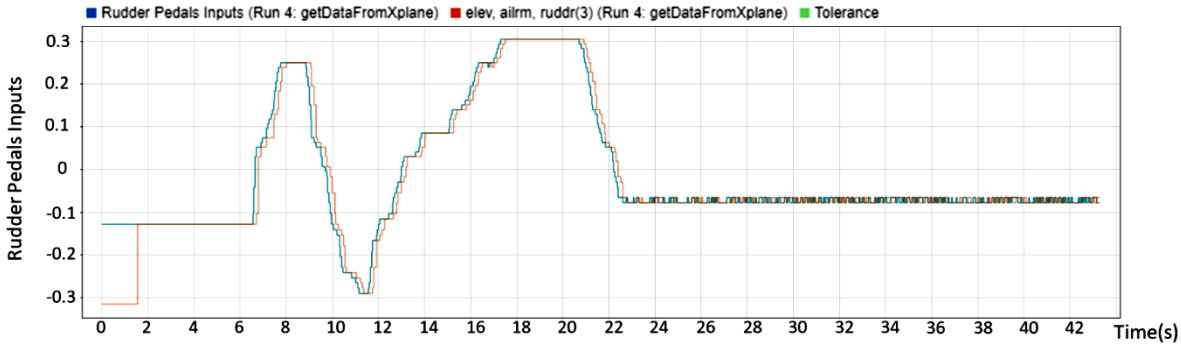


Fig. 10 Rudder Pedals Inputs

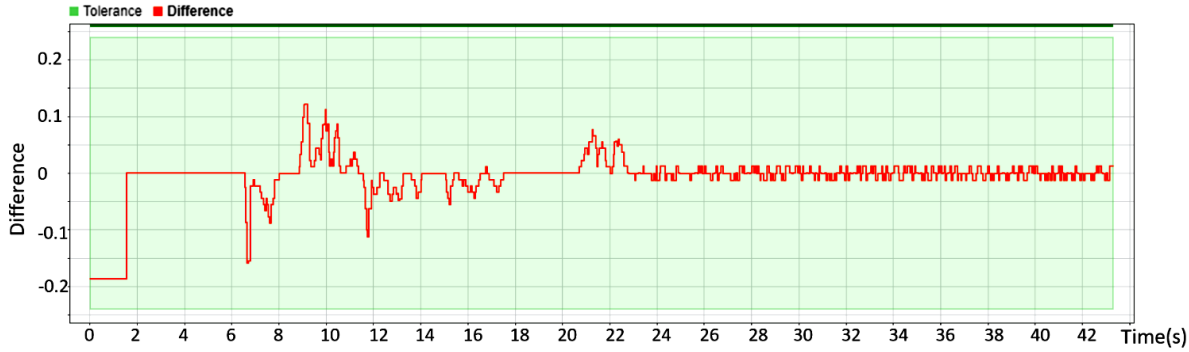


Fig. 11 Rudder Pedals Signals Difference

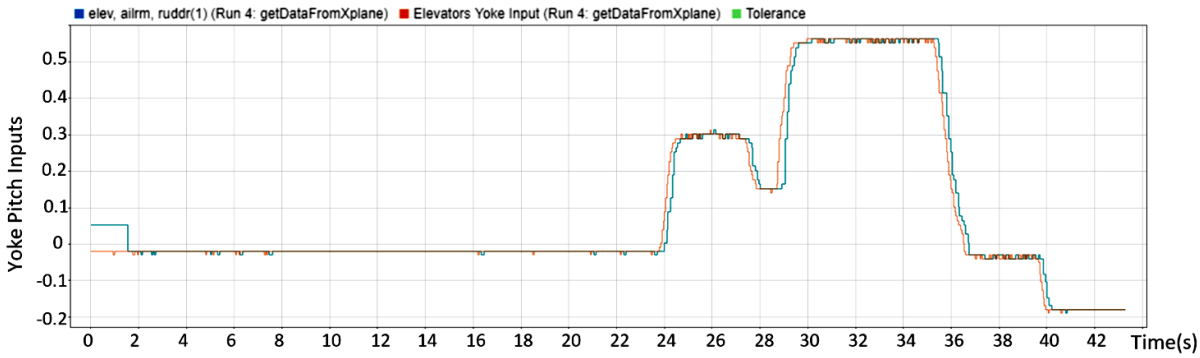


Fig. 12 Yoke's Pitch Inputs

Figure 13 shows the response difference between the two signals, with a small tolerance reaching 20%. For the third control parameter, the yoke's roll level which takes place after the departure by seconds. Figure 14 shows the two signals one coming from the yoke's microcontroller and one from X-plane. As shown in figure 15, the difference between the signals for this control parameter has been the lowest with tolerance reaching 8.5%.

Table 3 shows a summary of the minor tolerances obtained at the worst case during the interfacing process.

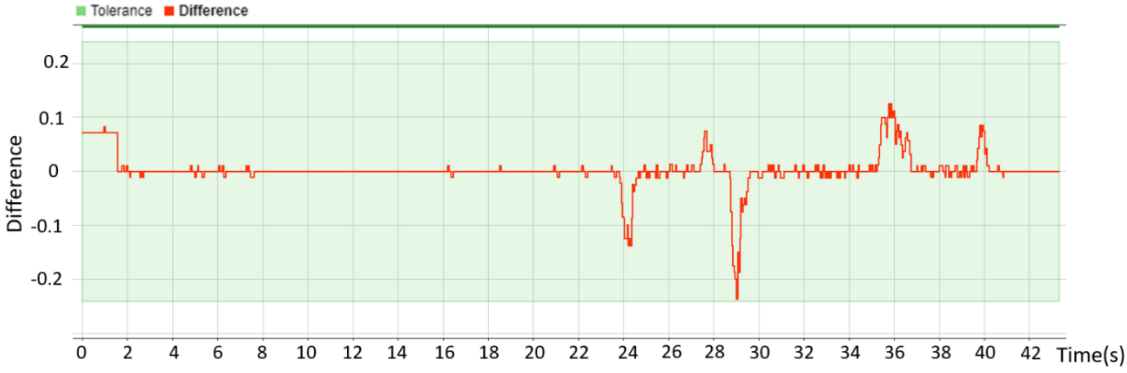


Fig. 13 Yoke's Pitch Signals Differences

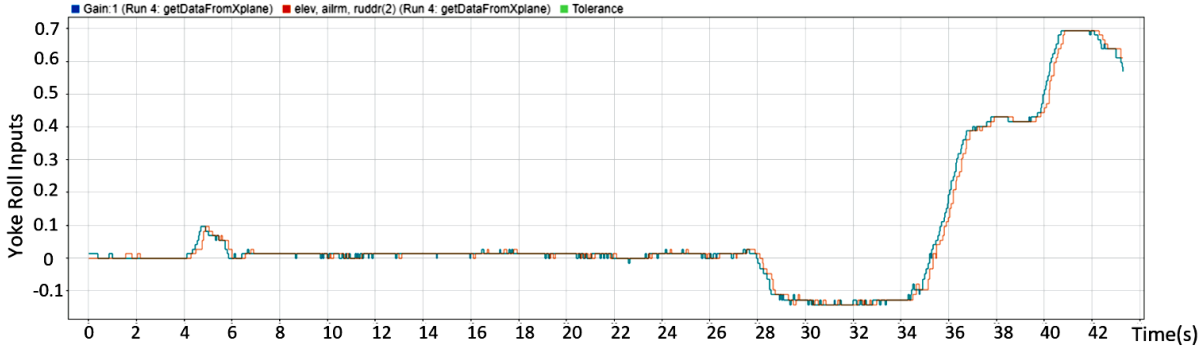


Fig. 14 Yoke's Roll Inputs

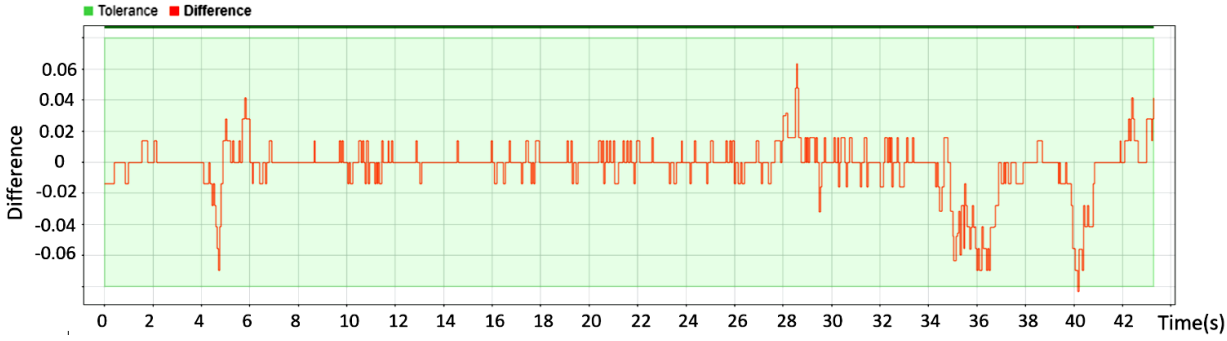


Fig. 15 Yoke's Roll Signals Differences

Table 3. Interfacing Tolerances

Input	Tolerance %
Rudder Pedals Inputs	16
Yoke Pitch Input	22
Yoke Roll Input	8.5

5.2. Motion Cueing Analysis

The tuning of the relevant filter parameters of the classical motion cueing algorithm requires extensive trial-and-error oriented testing at different stages of hardware integration. The response of the developed algorithm, with the current parameters, must be observed to determine whether the achieved results are promising for the continued development and use.

5.2.1 Experimental Setup

In order to find potential errors or areas that require fine tuning of filer parameters, a repetitive flight scenario is then required. This is implemented through the built-in situations feature in X-plane. Table 4 shows the scenario characteristics, while Fig. 16 shows the considered trajectory or the proposed scenario.

Table 4. Motion Cueing Experiment: Flight Scenario Characteristics

Aircraft Model	Cessna 172sp
Location	Cairo international Airport
Weather conditions	Clear / wind speed: 0.0 kts
Trajectory	Stadium shaped

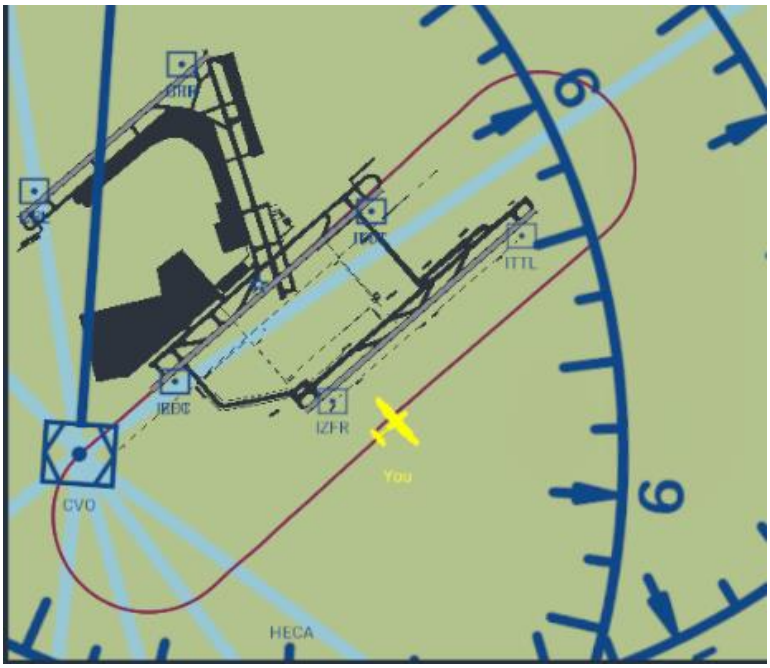


Fig. 16 Motion Cueing Experiment: Flight Scenario Trajectory

5.2.2 Results

Figure 17 illustrate the outputs of the motion cueing algorithm in the x direction. The platform acceleration resembles the input acceleration in terms of the signal form but is scaled down to be possible to implement on a motion platform. In addition, it is also possible to see the effect of the washout filter during all the acceleration spikes were the actual acceleration of the aircraft shows no decline while the filtered acceleration seeks to return to zero after every manoeuvre. This is why the platform acceleration will always be around zero. The same can be noticed when looking at the change in platform position over the runtime of the scenario, which can be seen to closely replicate the change in the acceleration but also maintaining the condition of returning to platform's neutral position.

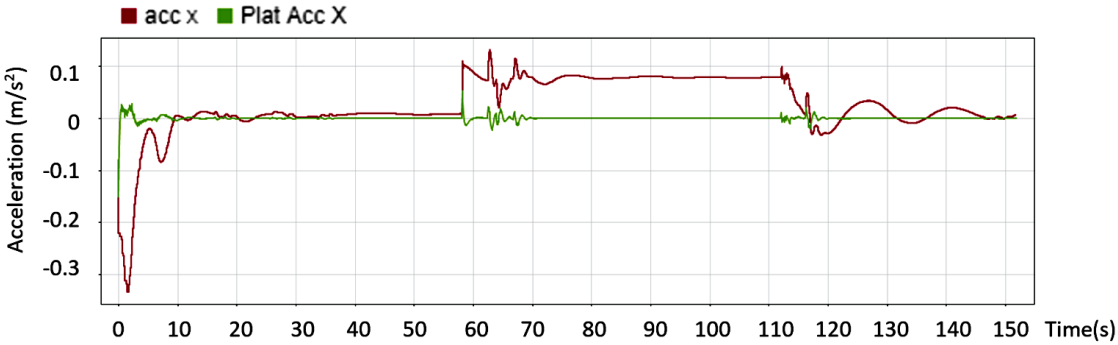


Fig. 17 Aircraft Acceleration vs. Platform Acceleration (Surge)

As for the rotational angle output of the platform, it is also consistent with what is expected. The effect of the low-pass filters is clear. The effect of the tilt coordination is also present in maintaining the platform pitch, roll, and yaw angles, Fig. 18. Platform angles are at or around a limited value to simulate the continuous change in translational acceleration, Fig. 19.

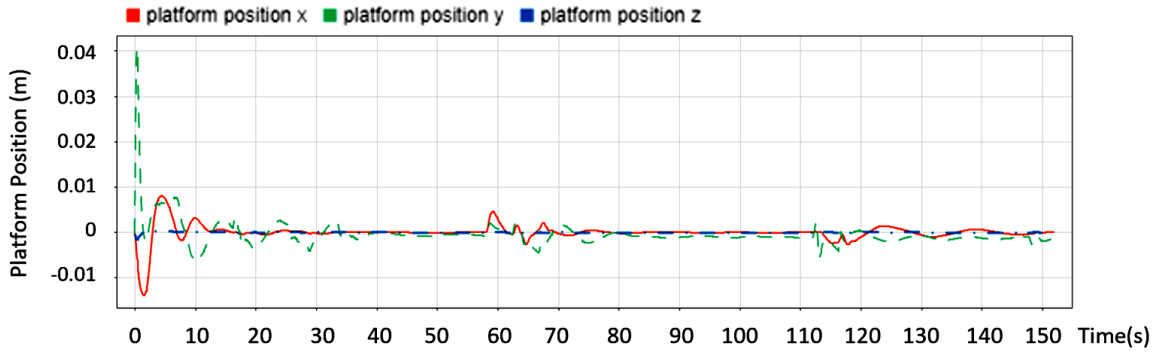


Fig. 18 Change in Platform Positions

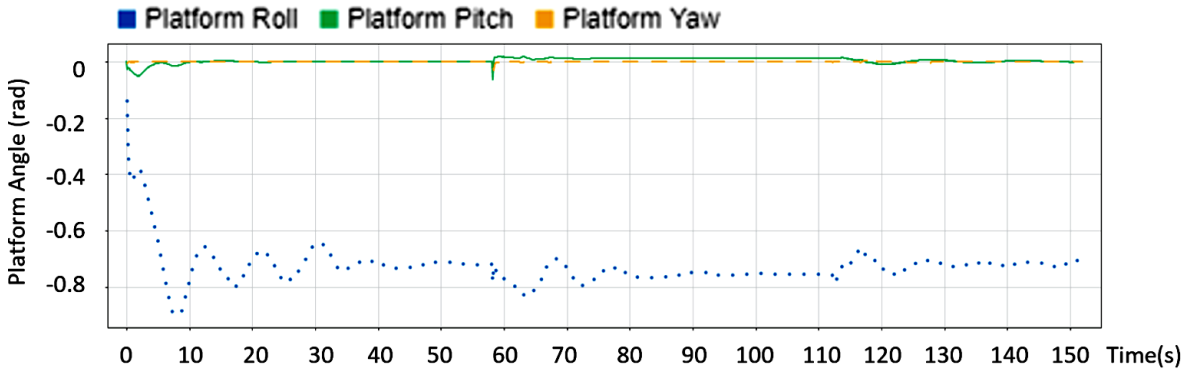


Fig. 19 Change in Platform Angles

5.2.3 Determination of Worst-Case Accelerations and Angular Velocities

After observing the behaviour of the motion cueing algorithm using the created scenario, it is now necessary to extensively test the algorithm in all situations in order to determine the worst-case acceleration around which the algorithm will be tuned. For this experiment, the same Cessna 172sp aircraft is flown in different scenarios such as take-off and landing, as well as several aerial manoeuvres. The results are illustrated by table 5.

It should be noted that the tuning of the motion cueing parameters takes place on the motion platform during its testing.

Table 5 Maximum Accelerations and Angular Rates

Quantity	Maximum Magnitude
Translational Acceleration-x (Surge)	$-10.5\ m/s^2$
Translational Acceleration-y (Sway)	$-38\ m/s^2$
Translational Acceleration-z (Heave)	$-4.1\ m/s^2$
Roll Rate	$\pm 0.5\ rad/s$
Pitch Rate	$\pm 0.5\ rad/s$
Yaw Rate	$\pm 0.2\ rad/s$

6. Conclusions

Motion cueing algorithms, based on the classical washout filter, was conducted, and used in the current flight simulator application. The simulator mechanical design and implementation of pilot control peripherals was undertaken. In addition, the interfacing and motion cueing models were tested in real-time experiments and yielded objectively satisfying results. The tolerances obtained from the interfacing process reached 15% error for the rudder pedals inputs, 22% error for the yoke pitch inputs, and 8.5% error for the yoke roll inputs, noting that these errors are the maximum errors detected during the simulation. Hence, the interfacing test using the implemented hardware peripherals and the implemented communication model provided satisfying results with a maximum error tolerance of 22% due to slight time delays but never actually affected the simulation fidelity.

Moreover, the worst-case accelerations and angular rates were determined and will be used to tune the algorithm accordingly. The maximum rates determined was $-10.5\ \mathrm{m/s^2}$ for the surge acceleration, $-38\ \mathrm{m/s^2}$ for the sway acceleration, $-4.1\ \mathrm{m/s^2}$ for the heave acceleration, $\pm 0.5\ \mathrm{rad/s}$ for the roll and pitch rates, and $\pm 0.2\ \mathrm{rad/s}$ for the yaw rate.

7. References

[1] M. A. Nahon and L. D. Reid, “Simulator motion-drive algorithms: a designer's perspective,” *Journal of Guidance Control Dynamics*, vol. 13, no. 2, pp. 356-362, 1990.

[2] F. Rehmatullah, “Model Predictive Control Based Motion Drive Algorithm for a Driving Simulator,” University of Toronto, Master's Thesis, 2017.

[3] R. V. Parrish, J. E. Dieudonne, R. L. Bowles and D. J. Martin Jr., “Coordinated adaptive washout for motion simulators,” *J. AIRCRAFT,* vol. 12, no. 1, pp. 44-50, 1975.

[4] D. Ariel and R. Sivan, “False cue reduction in moving flight simulators,” *IEEE Transactions on Systems, Man, and Cybernetics,* Vols. SMC-14, no. 4, pp. 665-671, 1984.

[5] R. Sivan, J. Ish-Shalom and J.-K. Huang, “An optimal control approach to the design of moving flight simulators,” *IEEE Transactions on Systems, Man, and Cybernetics,* Vols. SCM-12, no. 6, pp. 818-827, 1982.

[6] L. Nehaoua, H. Arioui, H. Mohellebi and S. Espie, “Motion cueing algorithms for small driving simulator.,” in *IEEE International Conference In Robotics and Automation (ICRA06),* Orlando,Florida,United States, 2006.

[7] M. Dagdelen, G. Reymond, A. Kemeny, M. Bordier and N. Maı¨zi, “Model-based predictive motion cueing strategy for vehicle driving simulators,” *Control Engineering Practice,* vol. 17, no. 9, pp. 995-1003, 2009.

[8] N. J. Garrett and M. C. Best, “Model predictive driving simulator motion cueing algorithm with actuator-based constraints,” *International Journal of Vehicle Mechanics and Mobility,* vol. 51, no. 8, pp. 1151-1172, 2013.

[9] B. Etkin and T. Teichmann, Dynamics of Flight: Stability and Control, vol. 9, Physics Today, 1959.

[10] M. Adamski, “Ergonomic And Psychomotor Conditions of Pilot's Work Environment,” 2020.

[11] N. Selim, A. Masoud and A. B. Aly, “Driver-in-the-Loop for computer-vision based ADAS testing,” 2021.

[12] Y. Amr, A. Mohamed and A. B. Aly, “Robust Lane Departure Warning System for ADAS on Highways,” 2022.