Realization of a Desktop Flight Simulation System for Motion-cueing Studies

Regular Paper

Berkay Volkaner¹, S. Numan Sozen¹ and V. Emre Omurlu¹*

¹ Mechatronic Engineering Department, Yildiz Technical University, Istanbul, Turkey
*Corresponding author(s) E-mail: eomurlu@gmail.com

Received 02 July 2015; Accepted 20 March 2016

DOI: 10.5772/63239

© 2016 Author(s). Licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract

Parallel robotic mechanisms are generally used in flight simulators with a motion-cueing algorithm to create an unlimited motion feeling of a simulated medium in a bounded workspace of the simulator. A major problem in flight simulators is that the simulation has an unbounded space and the manipulator has a limited one. Using a washout filter in the motion-cueing algorithm overcomes this. In this study, a low-cost six degrees of freedom (DoF) desktop parallel manipulator is used to test a classical motion-cueing algorithm; the algorithm’s functionality is confirmed with a Simulink real-time environment. Translational accelerations and angular velocities of the simulated medium obtained from FlightGear flight simulation software are processed through a generated washout filter algorithm and the simulated medium’s motion information is transmitted to the desktop parallel robotic mechanism as a set point for each leg. The major issues of this paper are designing a desktop simulation system, controlling the parallel manipulator, communicating between the flight simulation and the platform, designing a motion-cueing algorithm and determining the parameters of the washout filters.

Keywords Stewart Platform, Motion Cueing, Washout Filter, Flight Simulator

List of Symbols

| Symbol | Description |
|--------|-------------|
| $s$    | Frequency domain |
| $P$    | Position of centre of gravity of the top plate |
| $\omega$ | Filter cut-off frequency |
| $B$    | Pos. of centre of gravity of the bottom plate |
| $\zeta$ | Damping ratio |
| $\rho^\text{P}_i$ | Position vector of upper connection point of a leg w.r.t. mobile plate |
| $\rho^\text{B}_i$ | Position vector of lower connection point of a leg w.r.t. base plate |
| $\Phi_{x,y}$ | Tilt coordination around x and y axis |
| $\lambda_i$ | Offset angle of the platform leg ports |
| $g$    | Gravitational vector |
| $R$    | Rotational transformation matrix |
| $G_s$  | Semi-circular canal transfer function gain |
| $J$    | Jacobian matrix |
| $\tau$ | Time constants for semi-circular canal transfer function |
| $K_p$  | PID proportional coefficient |
| $K_i$  | PID integral coefficient |
| $K_d$  | PID derivative coefficient |
Flight simulators are necessary for studies related to vehicle design and tests, and simulator systems help us understand pilot behaviour that is close to reality. Flight simulators provide an economical and secure way of testing new technologies for implementing in aircraft. Several universities and industrial laboratories are undertaking research for new generation prototypes and vehicle dynamic models.

Simulators create a realistic flight feeling using motion information feedback [1], and are completely reliable, realizable and easy to implement. It is proposed to increase this reality by improving the software, hardware and physical abilities of simulators. To get closer to the real flight feeling, it is important to employ a robust motion-cueing algorithm [2].

Simulators are generally human in the loop (HITL) processes and include an intensive code algorithm. For instance, in a Level-D simulator there are numerous code arrays, which are expensive software systems. Because of their complex structure and high cost, simulator design is quite a difficult task [3].

In simulator systems, examining which factors cause the differences between real and virtual data is needed. These factors are generally separated into three categories:

- a. human motion perception (vestibular) systems
- b. the problems that may occur in visual flight simulation
- c. audio feedback

The questions possible in applications targeting human motion perception are:

i. What does human perception depend on in real-time applications?

ii. How is it possible to improve human perception by changing the parameters of a flight simulation system? [4]

Human motion perception systems are the basis of motion perception studies. Humans sense motion through the vestibular system located in the inner ear, which consists of semi-circular channels sensing angular motion and otoliths sensing translational motion. Mathematical models of the vestibular system were first represented in [5] and simplified transfer function models were presented in [6]. Then simulation studies became possible.

The otoliths are the human sensory system for translational accelerations and are stimulated by head position. There are three semi-circular canals for sensing angular acceleration or rotational motion. Corresponding signals are sent to the neural system through nerve fibres by the otolith organs’ receptor cells and semi-circular canals. The utricle and saccule are the otolith organs in the ear. The utricle senses horizontal motion and the saccule senses vertical motion. The vestibular system is associated with other organs responsible for sensing motion and it plays a major role in the balance of a human body [7, 8].

The otolith organs sense acceleration and the effect of gravity on the tilt of the pilot’s head. Since the otoliths, acceleration and tilt are associated with one another, this term is called tilt coordination and it improves the accuracy of any simulation. The tilt effect is performed by passing the aircraft translational acceleration through a low-pass filter. Additional low-pass filters are added into the motion-cueing algorithm in the pitch/surge and roll/sway modes [9] to integrate the tilt coordination into the motion-cueing algorithm.

Visual scaling results in differences in image perception. The optimal level of visual scaling affects the image perception positively. Changes in visual feedback were studied under different visual scale factors in [10].

Additionally, image depth and slope play a significant role in motion perception. Studies performed by Jamson confirmed that to create an accurate motion perception, the horizontal size of the image needs to cover 120°. In [10], the pilot was asked to move at a certain speed and simulation accuracy was tested.

Naturally, visual feedback in a simulation plays a significant role in motion perception. Image workflow, resolution or deflections in frequency adjustment create problems.

Initially, it was discovered that the image a pilot senses is based on differences in the optical workflow, which is a dynamic model the brain uses to solve a specific motion and helps the pilot predict absolute velocity, distance and relative spatial range. This image signal is effective for understanding major tasks like velocity of the vehicle, distance from objects and lateral control [4].

Audio feedback is less effective in motion perception. Sound is based on simulation acoustics and the number of
sounds supplied. Sound-emitting objects in the virtual environment make it easier to get an accurate space. Research has shown that sound is not effective in velocity control and it has been proved that rotation, lateral and yaw motion play a major role in sound vibration [4].

Inertial cues are very important for simulators. Without them, the pilot has no information about aircraft acceleration or rotation. Unlike real vehicles, motion platforms have limited workspace for rotation and acceleration. Motion-cueing algorithms are called washout filters and can be classified into several categories. These filters produce motion commands to be sent to the motion platform [11].

There are four algorithms in the literature for motion cueing:

1. classical approach
2. optimal algorithm
3. adaptive base algorithm
4. model-predictive approach

The classical algorithm was founded in the NASA Ames Research Center by Conrad and Schmidt in 1969 and Conrad, Schmidt and Douviller created the rotating coordinate filter in 1973. Generally, the classical approach consists of a combination of high- and low-pass filters in which the cut-off frequency and damping ratio are initially set. While examining the acceleration effect, integrating the gravitational effect into the acceleration change in the vertical direction is also an important issue. First, translational accelerations and angular velocities are scaled and angular velocities are passed through high pass filters. To return the simulator to its initial position, second- and third-order filters are used. If the effects are scaled and limited, then just the second-order filter is sufficient [12].

Because of the workspace limitations of a simulator, an adaptive algorithm was developed by Parrish, Dieudonne, Bowles and Martin in 1975, which is implemented in the frequency domain to eliminate the limited workspace problem. In this algorithm the coefficients are different [13] and motion perception is considered a tracking problem [14]. The acceleration that is obtained from the simulator should be tracked [15].

By applying an adaptive filter in a classical algorithm, the adaptive classical motion-cueing algorithm was developed. The adaptive gain is obtained by minimizing a cost function [12] and the high-pass filter gain is changed. The purpose of this algorithm is to minimize the cost function. Defusing the wrong cues at the actuator extensions and providing a smooth simulation environment is an advantage of the adaptive algorithm but due to the varying structure of the algorithm, there are sometimes gaps in motion perception. Adaptive motion-cueing structures also provide dynamic stability to the system in case of exposure to any distorting effects like parameter changes [16].

In [17], the developed adaptive algorithm structure was also examined. This structure was implemented as an alternative method to keep the simulator in the limits of the workspace. It has a single DoF motion-limit calculator block, which calculates the kinematics and the central point coordinates of the platform together. Corner frequency, damping ratio, gain, cost and adaptive step size as filter parameters are adjusted to keep the translational motion in the limited workspace. Initial cost values are determined based on experience. After the classical filter design, implementing a trial-and-error method is sufficient to determine the necessary parameters [18].

There are several studies comparing use of classical algorithm on three, six and eight DoF simulation platforms. On three DoF platforms, the translational feedback of the system was quite good. On six DoF platforms, motion was felt more intensively because of its freedom; eight DoF platforms showed similar performance [19].

The results of comparing platforms with different DoF are:

- The rotation effect must as small as possible.
- If the rotation coordination effect is unavoidable, the pilot’s head must be taken as the reference point.
- A small washout corner frequency must be chosen.
- Filter parameters must be chosen so to decrease the washout signal levels.

To handle the phase delay between the visual simulation and the platform, the software must be flexible and cost effective. There are also methods of compensation available (i.e., fuzzy logic systems, adaptive models and nonlinear solving) but most require closed-loop control [20].

The model-predictive algorithm is a novel motion-cueing algorithm approach and is presented in [21]. Model predictive control (MPC) is an advanced control technique where the provided control actions are optimized according to a determined system model as well as set of constraints on state evolution, input and input rates. MPC computes the control action by solving a finite horizon open-loop optimal control problem. Finally, by using the current state of the system for the initial state of the plant, the optimization process gives an optimal control sequence [11]. Using an internal model of the plant in MPC is thought of as a limitation, but it can also be considered an advantage for allowing the controller to understand system dynamics [22].

The MPC algorithm structure was tested in a Loughborough simulator and the results of MPC and the classical algorithm were compared. The Loughborough simulator is an electrically actuated Stewart platform. It has sensory feedback, three computer screens, a wheel torque feedback system and three speakers. The first two screens create the image and the last provides a real-time simulation appli-
Actuators of the simulator in the study had 600 mm stroke and ±20° angular motion capabilities.

In some cases, motion-cueing algorithms give good results but in others actuators may reach the maximum limits, which may disrupt the system that the adaptive algorithm has nonlinear filters for [23].

To reduce the cost function, the optimal control theory including the human vestibular system model was implemented by Sivan, Ishsalom and Huang in 1982. The optimal control theory is based on a problem of how to control the system as close to the desired value of specified vestibular system dynamics and constraints [24]. The linear motion perception model was suggested by Hosman and Van Der Vaart in 1981. In this algorithm, the Riccati equation is adjusted in real time using Newton Raphson method. It is observed that the optimal motion-cueing algorithm is more effective than the classical and adaptive algorithms [25].

Many types of washout filters are studied in the literature. In [26], the human vestibular system-based senseless manoeuvre optimal washout filter was proposed to decrease pilot sensation errors.

The human motion perception threshold values are presented and the platform is similar to an inverted pendulum.

The control strategy is separated into two controls: a conventional control such as PID and an advanced control such as MPC [27].

Studies on motion perception are based on linear filtering and optimal control. Recently, studies on the MPC algorithm have been performed and were associated with optimal control. It was observed that the MPC algorithm is more effective than the classical algorithm for similar conditions. In the MPC algorithm, the acceleration obtained from the simulation is converted to the acceleration sensed by the motion perception system. This acceleration value is used as a reference for operating the MPC algorithm. The MPC signals are obtained and transferred to the simulator. In the literature, the MPC algorithm was designed, separating the problem into two. Actuator lengths were used instead of Cartesian quantities to keep the platform inside the maximum limits of the workspace [28].

Pouliot suggested two approaches in looking at motion [29], the comparison of:

- simulator and air vehicle in terms of angular velocity and some produced forces; and
- angular velocity and jerk.

In [29] suggested approach is to compare:

- translational acceleration or angular velocity of simulator with vehicle data; or
- rotational acceleration or angular velocity with vehicle data.

In this study, a desktop flight simulation system is introduced with components including flight simulation software, a desktop-parallel robotic manipulator and a motion-cueing system. Human motion perception is summarized. Inverse/forward kinematics of the system and the real-time solution is explained. The communication problem between the simulation and the platform is addressed and, finally, the results from the hardware are discussed. The hardware used in the simulation and the real-time experiments is summarized in Figure 1.
c. Motion-perception software
   • Motion-cueing algorithm
   • Human perception model

2.1 Flight simulation software
Simulators have a visual interface on which the simulation of aircraft motion is performed. In this study, FlightGear 3.4 flight simulation software is used, which is an open-source software including several aircraft models. Because of the heavy workload of flight simulation, separate processors are allocated on separate computers to generate flight data, control signals and visual feedback of the flight. An HP z230 workstation computer with eight GB RAM and Nvidia NVS510 video card with four display ports are used to obtain uninterrupted visual flight feedback.

2.2 Motion simulator
The system is on a 6x6 parallel desktop manipulator and is driven by linear DC motors (Linmot PS01-23X80 / 0150-1201). Linear motors have a motion capacity of 110 mm; multi-axis motion control of the motors is performed via Linmot E210-VF model motor drivers, Figure 2. Figure 3 illustrates the joystick used and the flight simulation software.

2.3 Motion cueing
In flight simulators, all linear actuators are driven simultaneously to move the six DoF (x, y, z—roll, pitch, yaw) parallel manipulator to any desired position. Required positions of actuators are recalculated at each sampling time and the position error of the legs is kept to a minimum. The major issue in flight simulators is that the software has infinite space and platform has a limited one. A washout filter used in the motion-cueing algorithm eliminates this issue, which constrains the position data the platform receives and, most importantly, attempts to move the platform to its initial position under the driver’s perception of motion level.

The motion perception algorithm consisting of high-pass and low-pass filters is applied to reduce the translational acceleration and angular velocity values obtained from the FlightGear flight simulation software via a communication protocol to an acceptable limit of the platform workspace. Translational accelerations are filtered from high-frequency data and noise using a third-order (first-order high-pass washout and second-order low-pass washout) filter and angular velocities are filtered from high-frequency data and noise by a second-order washout filter.

As shown in Figure 4, the translational motion channel is responsible for simulating the aircraft translational motion. The channel’s input is the translational acceleration of the aircraft in the flight simulation software. The rotational motion channel simulates the aircraft rotations. Its input is the angular velocity of the aircraft. The tilt-coordination channel uses a tilt-coordination algorithm to estimate aircraft inertia along x and y axis. The output of this channel is summed with the rotational motion channel establishing the final rotational cue.

Filtered signals are transmitted to motor drivers, taking the double integrals of translational acceleration inputs and the because the internal control algorithms of linear motor drivers are extensively active in velocity mode, the force mode of drivers is selected so the authors can apply external control algorithms effectively. Position data are obtained via internal linear encoders of the motors with a sensitivity of 10 µm. Platform parameters are listed in Table 1.

![Motion system block diagram](image)

**Figure 2.** Motion system block diagram

![Joystick and Flight simulation software](image)

**Figure 3.** (a) Flight simulation software (b) joystick (Thrustmaster Hotas Warthog)

| Parameter                  | Value       |
|----------------------------|-------------|
| Top plate weight           | $M_u$ 1,387 kg |
| Motor shaft weight         | $m_u$ 0.135 kg |
| Motor body weight          | $m_d$ 0.44 kg |
| Top plate radius           | $r_p$ 0.15 m |
| Bottom plate radius        | $r_s$ 0.175 m |
| Motor inductance           | $L_a$ 1.5 mH |
| Motor resistance           | $R_a$ 10 ohm |
| Motor force coefficient    | $K_t$ 11 N/A |

**Table 1.** Parameters of 6x6 parallel manipulator mechanism
single integral of the angular velocity inputs, creating position information via the platform’s inverse kinematics. Platform velocity is kept lower than the human perception level during the washout process. Thus the pilot does not feel the washout motion presence. The combination of movement creates a real flight feeling. As shown in Figure 5, the classical algorithm has a simple structure, yet it is not adaptive due to its constant parameters.

In the algorithm, the first-order high-pass filter is as follows:

\[ \frac{s}{s + \omega} \]  

\( \omega \) is the filters’ corner frequency. In this experimental study, the factor is chosen according to the system performance and platform workspace boundaries.

2.3.1 Tilt-coordination effect

Figure 6 illustrates the tilt effect used to simulate translation acceleration using the rotational motion of the platform while the platform is motionless. Since motion simulators tend to have limited workspace, it is necessary to simulate translational acceleration using the Earth’s gravity and rotational motion of the platform when the platform is motionless. In a motion-cueing algorithm, the tilt coordination formulation can be shown as:

\[ \varphi_{x,g} = \arctan \left( \frac{\dot{a}_x}{g} \right) \]  

In Figure 6, the aircraft accelerates along the x axis and is tilted about y axis. \( \varphi_x \) and \( \varphi_y \) are the tilt angles of platform. \( a_x, g \) shows acceleration in the x and y directions of the platform and g is the Earth’s gravity. Since the tilt effect is a low-frequency motion, the simulation of translational acceleration can be easily simulated by using the Earth’s gravity [31]. However, high-frequency acceleration is simulated by the translational motion of the platform. The axes of the aircraft are represented in Figure 7.

2.3.2 Human motion perception

The human perception system is the basis for human perception studies. The vestibular system is shown in Figure 8. The dynamics between sensed angular velocities, \( \omega^i \), and the input angular velocities, \( \omega_i \), at each semi-circular canal with stimulus accelerations are given below. \( \tau_v, \tau_d, \tau_s, \tau_L \) are the time constants and \( G_s \) is the gain of the dynamic system.

In the transfer function between translational acceleration sensed in the ear, \( \dot{a}_x \), and stimulus acceleration, \( a_x \), 1/ \( \tau_L \) is
\[
W(s) = \frac{\tilde{a}(s)}{a(s)} = \frac{K_{oo}(s + a_0)}{(s + b_0)(s + b_1)}
\]  
(6)

Index \(i\) shows the rotational motions in Euler angles (roll, pitch, yaw). Human motion perception system parameters are shown in Table 2 and human motion perception threshold parameters are shown in Table 3.

A human detects acceleration and angular motion through the inner ear’s vestibular system. Because of insensitivity at low speeds, motion perception algorithms simulate the acceleration of the platform and attempt to return the platform to its initial position at these low speeds. When high frequency motion signals move the platform, low frequency motion signals are transformed into angular motion. The major issue in this motion is the loss of the acceleration’s low-frequency content. To eliminate this issue, the idea of tilt coordination was developed. The gravitational acceleration is \(g = 9.78033 \text{ m/s}^2\) which is the \(z\) component of the gravity vector. If an amount of roll motion is given as input to the simulator, the \(x\) component of the gravity vector can be adjusted to the level of the pilot’s motion perception.

### 3. Kinematic Analysis of Six DoF Parallel Mechanisms

As in serial robots, kinematic analysis of parallel mechanisms should be examined in terms of inverse and forward kinematics. Inverse kinematic analysis helps to find the leg lengths, \((L_1, L_2, L_3, L_4, L_5, L_6, L_7)\), given the position and orientation of the end effector when the position and rotation of top plate, \((x, y, z, \alpha, \beta, \gamma)\), is known. Forward

---

**Table 2.** Human motion perception model coeff.

| \(x\) | \(y\) | \(z\) | \( \hat{a}_x (s^{-1}) \) | \( \hat{a}_y (s^{-1}) \) | \( \hat{a}_z (s^{-1}) \) |
|---|---|---|---|---|---|
| 0,08 | 0,08 | 0,08 | 0,1 | 0,1 | 0,1 |
| 0,18 | 0,18 | 0,18 | 0,2 | 0,2 | 0,2 |
| 1,51 | 1,51 | 1,51 | 62,5 | 62,5 | N/A |

**Table 3.** Human motion perception thresholds

| Roll (deg/s) | Pitch (deg/s) | Yaw (deg/s) | \(a_x (\text{m/s}^2)\) | \(a_y (\text{m/s}^2)\) | \(a_z (\text{m/s}^2)\) |
|---|---|---|---|---|---|
| 2 | 2 | 1,6 | 0,17 | 0,17 | 0,28 |
Kinematics is the problem of finding the position and orientation of the top plate with respect to the fixed base when the leg lengths are known. Forward kinematics analysis is important in terms of all the conditions the platform may have in specific leg lengths. The contrast between serial and parallel robots continues here. When a specific trajectory needs to be followed, it is necessary to solve the inverse kinematics in real time. The forward kinematics solution of a parallel mechanism is not unique in that there are several possible configurations of leg lengths. In inverse kinematics, the solution is unique, which means one position/orientation of the top plate refers to certain leg lengths.

3.1 Inverse kinematic analysis

Inverse kinematic analysis for parallel mechanisms is simple to solve compared to forward kinematics. Inverse kinematics of a robotic system can be used to plan a specified trajectory that uses leg lengths by providing any specified position of the center of the platform top plate. The platform is shown in Figure 9.

\[ \{P\} \text{ demonstrates the position of the center of gravity of the top plate and } \{B\} \text{ demonstrates the position of the center of gravity of the bottom plate. } \Lambda \text{ can be defined as the angle between the forthrights drawn from the origin of axes, } \{B\}, \text{ to successive two corners of the fixed plate. } \lambda \text{ is the angle between forthrights drawn from the origin of axes, } \{P\}, \text{ to two successive corners of the moving plate.} \]

\[ \vec{p}_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \begin{bmatrix} r_i \cos(\Lambda_i) & r_i \sin(\Lambda_i) & 0 \end{bmatrix}^T \] (7)

\[ \vec{b}_i = \begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix} = \begin{bmatrix} r_b \cos(\Lambda_i) & r_b \sin(\Lambda_i) & 0 \end{bmatrix}^T \] (8)

\[ \vec{S}_i = -\vec{b}_i + \vec{p}_i \] (9)

The position of points \( P_i \) are defined by Equation (7) and position of \( B_i \) points by Equation (8). Leg length \( l_i \) can be obtained by calculating the magnitude of the \( \vec{S}_i \) vector and the open form of this vector is shown in Equation (9). \( i \) is the translational position of \( \{P\} \) with respect to \( \{B\} \). Rotation of the moving platform with respect to the \( \{B\} \) coordinate system is as follows:

\[ \vec{p}_i = \vec{p}_i \mathbf{R}^T \] (10)

\[ l_i = \sqrt{S_x^2 + S_y^2 + S_z^2} \] (11)

By integrating (9) into (11), the leg lengths are calculated according to the translation and rotation of the top plate. After successive regulations, the final inverse kinematics equation is obtained as follows:

\[ l_i^2 = x_i^2 + y_i^2 + z_i^2 + r_i^2 + r_i^2 + 2\left(r_i p_{ix} + r_i p_{iy}\right)(x - b_x) + 2\left(r_i p_{iz} + r_i p_{iy}\right)(y - b_y) + 2\left(r_i p_{ix} + r_i p_{iy}\right)(z - 2(xb_x + yb_y)) \] (13)

3.2 Forward kinematic analysis

There are several approaches on forward kinematics solutions in the literature. In this study, Newton–Raphson method is used and an equation in the form of \( F(x) = 0 \) is solved.

In this equation, \( x_0 \) is considered as initial point and if \( F(x) \) has a derivative at \( x_0 \), the tangent of the function at related point can be expressed as follows:

\[ Y = F(x_0)\left(x - x_0\right) + F(x_0) \] (14)

Next, \( x_i \) can be found by replacing \( x \) in the equation and zero instead of \( Y \).

\[ 0 = F(x_0)\left(x_i - x_0\right) + F(x_0) \] (15)

\[ x_i = x_0 - \frac{F(x_0)}{F(x_i)} \] (16)

\( x_i \) in Equation (16) will be closer to the solution. To achieve a more absolute solution with the same method, iterations are applied and solutions of \( x_5, x_5, x_5, x_5, x_5 \) can be obtained.
When the desired solution sensitivity is reached, iterations are terminated.

3.3 Implementation of forward kinematics

For the forward kinematics solution of the Stewart platform mechanism, Newton-Raphson method is applied as follows:

R is the rotational transformation matrix and the equation that will calculate leg lengths is defined in Equation (17). In this equation, x, y, z are translational positions of the mobile platform. \( r_p \) is radius of the circle passing through the joint centres of the top plate, \( r_b \) is radius of the circle passing through the joint centres of bottom plate. \( p_{ix}, p_{iy}, p_{iz} \) are coordinates of top plate–joint connections. \( b_{ix}, b_{iy}, b_{iz} \) are coordinates of bottom plate–joint connections. When the unknowns are \((x, y, z, \alpha, \beta, \gamma)\) and \(l_i\) is passed to the right side of the equation, (17) is obtained (Figure 10).

\[
F_i(X) = x^2 + y^2 + z^2 + r_p^2 + 2(r_{1x}P_{ix} + r_{1y}P_{iy}) (x - b_{ix}) + 2(r_{2x}P_{ix} + r_{2y}P_{iy}) (y - b_{iy}) + 2(r_{3x}P_{ix} + r_{3y}P_{iy}) (z - 2(xb_{ix} + yb_{iy}) - l_i^2 = 0
\]

\[
X = x, y, z, \alpha, \beta, \gamma \quad i = 1, 2, ..., 6 \tag{17}
\]

Because the function contains more than one variable, the derivative of the function used in Newton-Raphson method can be found by taking the partial derivative for each variable as follows:

\[
J_{ij} = \frac{dF_i}{dX_j} (X) \quad \forall \{i,j\} \in \{1..6\} \tag{18}
\]

Then, the unknowns are calculated by the following equation:

\[
X^{(n+1)} = X^{(n)} - [J(X^{(n)})]^{-1} \cdot F(X^{(n)}) \tag{19}
\]

The initial conditions are:

\[
X^{(0)} = (x_0, y_0, z_0, \alpha_0, \beta_0, \gamma_0) \tag{20}
\]

As a result of this iterative approach, a sufficiently accurate result is obtained. When approaching the solution, an \( \varepsilon \) value of sufficient accuracy should be defined as:

\[
|F_i(X)| \leq \varepsilon \tag{21}
\]

When Equation (21) is confirmed, the iterations are terminated and last \( X \) value is considered the solution. The flowchart of forward kinematics is in Figure 10.

4. Communication Software

To maintain signal transmissions between control/flight computers and the simulator in real time, the system’s communication is treated to preserve true motion feeling. Flight data is fed to the control computer to generate the necessary trajectories of the platform based on motion-cueing algorithms [32].

4.1 Communication and software

The mathematical model of the simulated aircraft is hidden in the visual simulation software, FlightGear, and the motion of the aircraft is commanded using an external professional joystick system. These mathematical models produce motion variables for the flight, which are transferred to the control computer. A generic protocol is used on the input/output port of the visual simulation software. The flight parameters produced by the flight simulation software’s mathematical models are transmitted via generic protocol as data packages and delivered into the platform control computer employing the TCP/IP protocol. During this process, data to be transmitted are defined by creating an XML script file. Thus, six parameters including translational accelerations of the simulated aircraft at

---

Berkay Volkaner, S. Numan Sozen and V. Emre Omurlu: Realization of a Desktop Flight Simulation System for Motion-cueing Studies
different pilot positions and angular velocities are transmitted to an external port (Table 4).

| Time  | \(\omega_x\) (rad/s) | \(\omega_y\) (rad/s) | \(\omega_z\) (rad/s) | \(a_x\) (m/s\(^2\)) | \(a_y\) (m/s\(^2\)) | \(a_z\) (m/s\(^2\)) |
|-------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|
| 19.74 | -0.08                | 0.13                 | 0.44                 | 9.26                 | -0.57                | -9.85                |
| 19.76 | -0.01                | 0.13                 | 0.47                 | 9.26                 | -0.62                | -9.85                |
| 19.77 | 0.08                 | 0.13                 | 0.5                  | 9.26                 | -0.65                | -9.85                |
| 19.79 | 0.18                 | 0.13                 | 0.53                 | 9.26                 | 0.68                 | -9.85                |
| 19.81 | 0.27                 | 0.13                 | 0.56                 | 9.26                 | -0.74                | -9.85                |

Table 4. FlightGear sample data output

The created XML script is kept in the visual simulation software’s setup folder and is run here in real time. The XML script is shown in Figure 11.

```
<PropertyList>
  <generic>
    <output>
      <line_separator>true</line_separator>
      <var_separator> </var_separator>
    </generic>
    <name>cran (sec)</name>
    <type>float</type>
    <format>4.2f</format>
    <node>/sin/time/elapsed-sec</node>
  </chunk>
  <name>roll angular velocity (deg/sec)</name>
  <type>float</type>
  <format>8.2f</format>
  <code>(orientation/roll-rate-deg)</node>
  <factor>0.3085</factor> <!-- to feet to meter -->
</chunk>
```

generic=socket.out,120,174.26.78.124,5555, tcp, mypr otocol.xml

Figure 11. XML script file and last expression

It is determined that a sampling frequency of 240 Hz is sufficient for visual simulation. Thus, the visual simulation external communication frequency is chosen to be 120 Hz based on the Nyquist criterion. Flight data obtained from the flight simulation software are transmitted to the defined port of motion control computer via a TCP/IP protocol (Figure 12). A server in an m-File is created to receive incoming data in a Matlab environment.

```
while 1
  if t.BytesAvailable > 0
    data = fread(t, t.BytesAvailable);
    sprintf('%s\n', str);
    try
      n = 1;
      for k = 1:8
        str = 'n';
        n = n + 1;
      end
      set_param('Model', 'SimulationCommand', 'update');
      pause(0.05);
      end
    catch
      fclose(t);
    end
end
```

Figure 13. Format variation code

4.2 Control

The parallel manipulator is controlled in two stages:

1. Force (current) control of linear actuator drivers
2. Individual leg position PID controllers implemented in a Simulink control model over the Q8 card

The overall structure of the control model is represented in Figure 14, in which each leg’s control signal is generated and fed separately. The output of the PID controller is an analogue input to the motor drivers as ±10 V, which is considered to be a current set point since the drivers are used in force-control mode. The linear actuator inner control model is shown in Figure 15. This input relationship is shown in Figure 16. Thus, ±10 V refers to ±3A in the driver controller, which results in force control over the linear actuator. Drivers of the linear actuators are set to provide auxiliary encoder output in 10 µ resolution (over the possibilities of 1, 5, 10, 20 and 50 µ), which provides the best performance for the setup used. Since set points regarding FlightGear data is compared with actual encoder readings, the difference between them is considered to be the error supplied to the controller structure. Flight Simulation by FlightGear 3.4 can be run in 5-10 Hz; 5 Hz is employed in this application because of the general acceptance by JAR standards for level D simulators.

Because of the nonlinearity and complexity of the platform kinematics, PID parameters are selected based on Table 6.
PID CONTROLLER COEFFICIENTS

| Parameter | $K_P$ | $K_I$ | $K_D$ | Filter |
|-----------|-------|-------|-------|--------|
|            | 600   | 23    | 8     | 115    |

Table 5. Platform mathematical model controller coefficients

| Parameter Increase | Rise Time | Overshoot | Settling Time | Steady-State Error |
|--------------------|-----------|-----------|----------------|--------------------|
| $K_P$              | i         | 1         | Small Change   | i                  |
| $K_I$              | i         | 1         | 1              | Great Reduction    |
| $K_D$              | Small Change | 1       | 1              | Small Change       |

Table 6. Coefficient properties of PID

5. Results

Washout filter parameters are chosen as in Table 7 using trial-and-error method in which 18 parameters are present and motion range, limited workspace, obtaining logical motion behaviour and the security of the platform are the considerations. By varying the parameters with small intervals, the motion of the platform has been observed and best results are chosen based on above criteria.

Simulation tests are performed on the desktop simulator system. Boeing 777-300ER model aircraft is prepared for departure from a specified airport in FlightGear 3.4 visual simulation software. The acceleration effect on the aircraft during departure is simulated as an amount of motion in the x-axis (forward) direction (Figure 17).

The aircraft moved through x axis with $4 \text{ m/s}^2$ acceleration and the platform moved 0.08 metres through x direction. The platform rotated $8^\circ$ about the y axis simultaneously because of the tilt coordination to simulate continuing forward acceleration using a gravitational effect. The results are as shown in Figure 18.

Figure 14. Simulink model of the simulator

Figure 15. Linear actuator driver control model

![Simulink model of the simulator](image1)

![Linear actuator driver control model](image2)

![Relationship between analogue input and supply current](image3)

![First-order high pass filter](image4)

![Second-order high pass filter](image5)

![First-order low pass filter](image6)

![x axis](image7)

![y axis](image8)

![z axis](image9)

![Pitch](image10)

![Roll](image11)

![Yaw](image12)

![Figure 16. Relationship between analogue input and supply current](image13)

![Figure 17. Motion at 20th second](image14)

![Figure 18. Motion at 20th second](image15)

![Figure 19. Motion at 20th second](image16)

![Figure 20. Motion at 20th second](image17)

![Figure 21. Motion at 20th second](image18)

![Figure 22. Motion at 20th second](image19)

![Figure 23. Motion at 20th second](image20)

![Figure 24. Motion at 20th second](image21)

![Figure 25. Motion at 20th second](image22)

![Figure 26. Motion at 20th second](image23)

![Figure 27. Motion at 20th second](image24)

In the second case, the aircraft performed roll motion and moved about the x axis through a CCW direction with an alpha angle of $-5^\circ$ (Fig. 19). Then the following motion about the clockwise direction with a $7^\circ$ alpha angle (Fig. 20) is performed. Results are as shown in Figure 21 and Figure 22.

In the third case, the aircraft descended with an angle of $10^\circ$ about the y axis (Fig. 23) and lifted its nose with $15^\circ$ about the y axis (Fig. 24). Results are as shown in Figure 25. In this motion, the platform moved through z axis for perceived g-force of aircraft. The platform moved -0.05 metres when the aircraft dived. Secondly, the platform moved 0.1 metres in a z direction when the aircraft lifted its nose.

Figure 16. Relationship between analogue input and supply current

Figure 17. Motion at 20th second

Figure 18. Motion at 20th second
The results of the simulation on a desktop simulator are considered to be satisfactory and the desktop system is proven to be more economic to use in trials of various washout filters. As a further analysis, the pilot’s perception of motion is to be analysed using the provided vestibular model and forward kinematics of the platform. Since the results obtained are based on a classical washout algorithm, further investigation is to be performed using extensive algorithms like adaptive, optimal and model predictive washout structures.
6. Conclusions and Future Work

This paper presents a novel and economical desktop flight simulator system. When compared to similar systems, it can be seen that the proposed system is more convenient for motion-cueing studies because it is simple and economic to apply developed motion-cueing algorithms.

The FlightGear package programme uses XML as a data protocol to provide communication between FlightGear and MATLAB via TCP/IP protocol so an XML script file is necessary for this system.

The development of other motion-cueing methods (optimal, model predictive, adaptive) is also being performed using this system. The main goal is to optimize the algorithm giving the best results to use in a full flight simulator system (FFS, Level D), which is already constructed and in use.

7. Acknowledgements

This article is a revised and expanded version of a paper entitled Experimental Motion Cueing Studies Employing Desktop Flight Simulation System presented at 14th International Conference on Circuits, Systems, Electronics, Control & Signal Processing (CSECS ‘15), Konya, 2015. This work is supported by the Ministry of Science, Industry and Technology of Turkey under grant 0563.STZ.2013-2 and by Scientific and Technological Research Council of Turkey, TUBITAK. The information provided in this article does not reflect the ideas of the Ministry.

8. References

[1] N.J. Garrett, M.C. Best, Evaluation of a New Body-Sideslip-Based Driving Simulator Motion Cueing Algorithm, Journal of Automobile Engineering, pp. 1433-1445, May 18, 2012.

[2] H. Elloumi, M. Bordier, N. Mažić, Optimal Control Scheme For A Driving Simulator, In Proceedings of the Second International Conference on Informatics in Control, Automation and Robotics, pp. 40-47, 2005.

[3] Z. Lei, J. Hongzhou, PC Based High Quality and Low Cost Flight System, International Conference on Automation and Logistics, Jinan, China, August 18-21, pp. 1017-1022, 2007.

[4] A. Hamish, J. Jamson, Motion Cueing in Driving Simulators for Research Applications The University of Leeds Institute for Transport Studies, PhD Thesis, November 2010.

[5] G.L. Zacharias, Motion Cue Models for Pilot-Vehicle Analysis, AMRL – TR – 78 – 2, Wright – Patterson Air Force Base, Ohio, May 1978.

[6] R.J. Telban, W. Wu, F.M. Cardullo, Motion Cueing Algorithm Development: Initial Investigation And Redesign of the Algorithms, National Aeronautics and Space Administration, Langley Research Center; CR – 2000 – 209863, 2000.

[7] A. Capustiac, B. Hesse, D. Schramm, D. Banabic, A Human Centered Control Strategy for a Driving Simulator, International Journal of Mechanical & Mechatronics Engineering IJMME-IJENS, Vol:11, No:01, February, 2011.

[8] J.A. Houck, R.J. Telban and F.M. Cardullo, Motion Cueing Algorithm Development: Human – Centered Linear and Nonlinear Approaches, Technical Report. CR-2005-213747, Nasa, 2005.

[9] Y.H. Chang, C.S. Liao, W.H. Chieng, Optimal Motion Cueing for 5 – DOF Motion Simulations via a 3 – DOF Motion Simulator, Control Engineering Practice, pp. 170 – 184, 2009.

[10] F. Colombet, D. Paillot, F. Merienne, A. Kemeny, Visual scale factor for speed perception, Journal of Computing and Information Science in Engineering, American Society of Mechanical Engineers (ASME), 2011, 11 (4), pp.041010-1 to 041010-6.

[11] B.D. Correia Augusto, R.J. Leal Loureiro, Motion Cueing in the Chalmers Driving Simulator: A Model Predictive Control Approach, Master of Science Thesis, Chalmers University Of Technology, Department of Signals And Systems, Report No. EX057/2009.

[12] L. Reid, A. Nahon, Simulator Motion-Drive Algorithms: A Designer’s Perspective, The University of Toronto, Vol. 13, NO. 2, 1990.

[13] M. Aminzadeh, A. Mahmoodi, Optimal Motion Cueing Algorithm Using Motion System Kinematics, European Journal of Control (4), pp. 363 – 375, 2012.

[14] L.D. Reid and M.A. Nahon, Flight simulation motion-base drive algorithms: Part 1 – Developing and Testing the Equations, UTIAS Report No. 296, CN ISSN 0082-5255, December, 1985.

[15] R. Sivan, J. Ish-Shalom, and J.-K. Huang, An Optimal Control Approach to the Design of Moving Flight Simulators, Systems, Man and Cybernetics, IEEE Transactions on Systems, Vol. 12, pp. 818-827, 1982.

[16] R.A. Stela, C. Miklos, L.C. Lihaciu, Adaptive Control of Pantograph – Catenary Interactions Force, WSEAS Transactions on Systems, Volume 13, pp. 708-719, 2014.

[17] Y. Yang, S. Zheng, Motion Drive Algorithm for Flight Simulator Based on the Stewart Platform Kinematics, School of Mechatronics engineering, Harbin Institute of Technology, Harbin, 150080, China, Key Engineering Materials Vols. 460-461 (2011) pp. 642-647, 2011.

[18] G. Reymond, A. Kemeny, Motion Cueing in the Renault Driving Simulator, International Journal of Vehicle Mechanics and Mobility, pp. 249–259, August, 2000.

[19] H. Sehammer, M. Fischer, G. Palmkvist, Motion Cueing for 3-6 and 8 Degrees of Freedom Motion...
[20] Z. Fang, G. Reymond, *Performance Identification and Compensation of Simulator Motion Cueing Delays*, RENAULT, Technical Center for Simulation, TCR AVA 0 13, 1, Avenue du golf – 78288 Guyancourt, France, Vol. 11, December 2011.

[21] M. Dagdelen, G. Reymond, A. Kemeny, M. Bordier, and N. Maizi, *Model-based predictive motion cueing strategy for vehicle driving simulators*, Control Engineering Practice, Vol. 17, pp. 995-1003, 9, 2009.

[22] M.L. Saidi, A. Debbeh, H. Arioui, M.S. Kermiche, and H.A. Abbassi, *Predictive control of motion platform in driving simulator*. Asian journal of information technology, Vol. 5, No. 2, pp. 133–138, 2006.

[23] L. Nehaoua, H. Arioui, S. Espie and H. Mohellebi, *Motion Cueing Algorithms for small Driving Simulator*, IEEE International Conference in Robotics and Automation (ICRA06), pp. 3189-3194, 2006.

[24] C. Carlota, S. Chá, *An Existence Result for Non-Convex Optimal Control Problems*, WSEAS Transactions on Systems and Control, Vol. 9, pp. 687-697, 2014.

[25] P. C.I. Huang, L.C. Fu, *A Time Varying Washout Approach for Flight Simulation Hexapod Motion System*, IEEE, Aviation University of Air Force Changchun, China, pp. 260-263, 2012.

[26] Chin-I Huang, Li-Chen Fu, *Senseless Maneuver Optimal Washout Filter Design with Human Vestibular Based (HVB) for VR-based Motion Simulator*, 17th IFAC World Congress (IFAC’08) Seoul, Korea, July 6-11, pp. 14755-14760, 2008.

[27] A.M. Yousef, *Model Predictive Control Approach Based Load Frequency Controller*, WSEAS Transactions on Systems and Control, Issue 7, Volume 6, pp. 265-275, 2011.

[28] M. Baseggio, A. Beghi, *An MPC Approach to the Design of Motion Cueing Algorithms for Driving Simulators*, 14th International IEEE Conference on Intelligent Transportation Systems Washington, DC, USA, October 5-7, pp. 692-697, 2011.

[29] I.A. Qaisi, A. Traechtler, *Human in the Loop: Optimal Control of Driving Simulators and New Motion Quality Criterion*, University of Paderborn, Paderborn, Germany, pp. 2235-2240, 2012.

[30] V.E. Ömürli, I. Yildiz, *Parallel self-tuning fuzzy PD+PD controller for a Stewart–Gough platform-based spatial joystick*, Arabian Journal for Science and Engineering, Vol. 37/7, pp. 2089-2102, 2012.

[31] S.H. Chen, L.D. Fu, *An Optimal Washout Filter Design for a Motion Platform with Senseless and Angular Scaling Maneuvers*. Proceedings of the American Control Conference, USA, pp. 4295 – 4300, June 30 – July 02, 2010.

[32] B. Volkaner, S.N. Sozen, H. Altuntaş, E.E. Kaya, V.E. Omurlu, *Experimental Motion Cueing Studies Employing Desktop Flight Simulation System*. Proceedings of the 14th International Conference on Circuits, Systems, Electronics, Control & Signal Processing (CSECS ’15), (pp. 24 – 33). WSEAS Press, Konya, 2015.