Chapter

A New Real-Time Flight Simulator for Military Training Using Mechatronics and Cyber-Physical System Methods

César Villacís, Walter Fuertes, Luis Escobar, Fabián Romero and Santiago Chamorro

Abstract

So far, the aeronautical industry has developed flight simulators and space disorientation with high costs. This chapter focuses on the design and implementation process of a low-cost real-time flight simulator for the training of armed force pilots using mathematical models of flight physics. To address such concern, the mathematical models of a Cessna type aircraft have been developed. This has been followed by a flight simulator, which operated with a new construction using a Stewart scale platform and operated by a joystick. Specifically, the simulator has been developed using an approximation of a physical cyber-system and a mechatronic design methodology that consists of mechanical, electrical and electronic elements that control the Stewart platform with three degrees of freedom. Based on software engineering, the algorithms of mathematical and physical models have been developed. These have been used to create an interactive flight simulator of an aircraft based on the Unity 3D game engine platform. The performance of the algorithms has been evaluated, using threads and processes to handle the communication and data transmission of the flight simulator to the Stewart platform. The evaluation of the developed simulator has been validated with professional pilots drilled with the Microsoft Flight Simulator. The results demonstrated that this flight simulator stimulates the development of skills and abilities for the maneuver and control of an aircraft.

Keywords: flight simulator, mathematical models of aircraft, flight training, Stewart platform, real time simulation, mechatronics, cyber-physical system, kinematic control

1. Introduction

Modern flight simulators meet two main aviation objectives: (1) to provide pilot training at the instructor's level, and at the student's level to learn to fly and to earn virtual flight hours that are useful for flying real aircrafts and (2) to simulate normal flight conditions, as well as adverse situations and spatial disorientation such as navigation instrument faults, power losses, loss of control of the aircraft, confusion illusion of references, illusion of the effect of black holes, among others, that would be dangerous and even catastrophic in a real flight; so, they must be well analyzed, controlled, and learned.

Real flight simulators with full movement generate movements and images where pilots feel an almost 100% level of realism of what would happen in a real plane. These simulators combine a series of technological aspects such as the Stewart platforms that reproduce real-time movements of the simulator software at hardware level and allow stimulating the visual and vestibular system of the pilots, reaching a maximum level of knowledge of various types of favorable and adverse situations and spatial illusions.

The main objective of this research was to design and build a flight simulator as a cyber-physical system, both at the software level and at the hardware level; based on mathematical models and programming algorithms that allow us to recreate a Cessna 172 type airplane in a virtual world and to prove its correct functioning. For that purpose, the Unity 3D gaming engine has been used as a developmental tool, together with the LabVIEW graphical programming environment to access the hardware and data information of the built-in Stewart platform with three degrees of freedom.

The flight simulator as a cyber-physical system composed of software and hardware was evaluated in terms of its functionality with a group of Ecuadorian aviation pilots who met basic training flight hours with the Cessna 172 and the ENAER T-35 Pillan, and also met virtual flight hours using a personalized license from Microsoft Flight Simulator.

The main contributions of this study were: (a) design of a mathematical model of front velocity, vertical velocity and lateral velocity of an airplane, as well as the application of the physical forces involved in an airplane such as lift, weight, thrust, and drag; (b) implementation of a dynamic Cyber-physical system, where the Mechatronic system was part of it and consisted of a software flight simulator programmed with the UNITY 3D framework. It allowed to include 3D objects such as terrains, buildings, airplanes, cities, sea, etc.; as well as, to model the cockpit in 3D as it looks in the reality and a Stewart platform (hardware) to scale with three degrees of freedom that reproduces the movements of the simulator plane in relation to the roll, pitch, and yaw, operated by a joystick.

This research has been organized as follows: Section 2 talks about the mechatronics systems design based in V-Model. Then, Section 3 explains about mathematical foundations of Stewart-Gough Platform with 3-DOF. Later, the flight simulator construction has been clarified in Section 4. Next, Section 5 presents the experimental results. Finally, Section 6 gives the conclusions and future work.

2. Mechatronics systems design based in V-Model

2.1 Philosophy

The philosophy of designing mechatronic systems has evolved over the years along with its definition, applications, and boundaries. Since its origins, mechatronics has tried to analyze systems holistically, with a synergic point of view integrating heterogeneous components (mechanics, electronics, and informatics) and subsystems to create more complex systems. This method requires a development that delivers products independently of their domains.

The V-Model allows the design and development of complex mechatronic systems with an interdisciplinary approach, where the VDI guideline 2206 can be applied to obtain systems that are more flexible and adaptable to the needs of users [1]. This model considers different factors, components, and the synergistic behavior of the different parts of a mechatronic product and its integration.

The implementation of mechatronic systems has a great reception in the military area. The development of vehicles is one of the most obvious examples due to the high variety of functions that must be fulfilled, according to [2] the idea of using mechatronic systems in the design of combat vehicles arrived with the twenty-first century when the basic architecture of electric and electronic components was developed, a clear example of these systems can be observed in [3, 4].

The framework given for that methodology has as main objective to generate a product based on the client requirements. In order to accomplish their requirements, the characteristic method to design and develop mechatronic systems is composed of four phases: (1) requirements phase—requirements and specifications analysis is customer oriented and defines the start and end of the project; (2) functional characteristics phase—the functions that are directly or indirectly visible by the user are defined; (3) design phase—the hardware and software components are defined, as well as the architecture of the system; (4) implementation phase—all the unit elements or system programming modules are developed [5]. Then, in the present study, this methodology contributed in the design and implementation of hardware for a simulator.

2.2 Cyber-physical system (CPS)

Cyber-physical system (CPS) is a new concept around the revolution of Information and Communications Technology (ICT) and Embedded Systems, and it refers to the integration of computing, data networks and physical processes to become intelligent objects that can cooperate with each other, forming distributed and autonomous systems. A CPS can include several disciplines related to software (Systems Engineering, Computation, Communication, and Control) and hardware (Industrial Engineering, Mechanical, Electrical, and Electronic) [4].

Nowadays, thanks to the internet of things, there is a greater development of products that have computing and communication. These products need an intimate coupling between the cyber and physical components and will be presented in the nano at large-scale world. CPS may be considered a confluence of embedded, real-time and distributed sensor systems as well as control [6, 7].

As it can be seen in the military [8] the modeling of present physical systems has several challenges, such is the case of fuel tanks in airplanes. Maintaining a correct monitoring of the level of fuel is a complex job since it requires the use of several sensors and the consideration of several external factors, but when generating a solution with CPS has advantages in the monitoring of it that helps to prevent accidents such as the case of Air Transat Flight on August 23, 2001; where due to a maintenance failure, the aircraft was left without fuel for the landing. This type of systems can be implemented in various systems of military vehicles such as jet aircraft, helicopters, etc.

By having a CPS, it is possible not only to monitor, but to implement an autonomous control; this can be done in any sector such as factories, transportation, aerospace, buildings and environmental control, process control, critical infrastructure, and healthcare. As indicated by [9], CPS projections will have a great impact in a wide variety of areas, and [10] states that Networked autonomous vehicles could dramatically enhance the effectiveness of the military and could offer substantially more effective disaster recovery techniques.

The biggest opportunity for implementing CPS is given by the opportunity to decrease costs and simultaneously increase capacity of sensors and actuators; in addition to the access to high capacity, smaller formed computing devices, wireless communication, increased bandwidth and continuous improvements in energy. The advantages and opportunities offered by the cyber-physical systems (CPS) have allowed innovation and improvement of the principles of engineering as Rajkumar mentioned in his research [9]. One of the main challenges that arises in the design and development of cyber-physical systems is the development of new methods of science and system design engineering to obtain a CPS that is compatible, reliable, integrated and synergistic in all the five-layers of functionality architecture with other cyber-physical systems. This research contributes to the design, development and implementation of a CPS following engineering principles.

2.3 Mechatronics and CPS

Naturally CPS and mechatronics systems are different, each one follows different goals, but its subtle differences characteristics make those systems complementary; since CPS consider the mechatronic system as an integral part of them. **Table 1** shows the differences between mechatronic systems and CPS.

According to the works of [11, 12], the cyber-physical systems are composed of two parts.

2.3.1 Software (Cyber)

This component has as areas of influence the Systems Engineering, Computation Engineering, Communication Engineering, and Control Engineering; all of these supported by the Application Lifecycle Management (ALM).

2.3.2 Hardware (Physical)

This component refers to the mechatronic itself, which has as areas of influence Industrial Engineering, Mechanical Engineering, Electrical Engineering, and Electronic Engineering; all of these supported by the Product Lifecycle Management (PLM).

Parameters	Mechatronics	CPS
Maintainability/availability	+	
Scalability		++
Stability		+
Robustness		++
Efficiency	+	+
Autonomous		++
Energy efficiency	+	++
Safety	++	++
Compactness	+	
Reliability	++	+
Accuracy	++	+
Communication	++	++

Table 1.
Differences between mechatronic systems and CPS.

DOI: http://dx.doi.org/10.5772/intechopen.86586

In this research, it has been designed as a complex system applying the approach of cyber-physical systems, where all the hardware components designed complied with Mechatronic processes and the software components designed complied with software engineering processes. Traditionally, practitioners of the mechatronic design philosophy had focused in the creation of a specialized architecture depending on the proposed solutions, while CPS are per se more distributed systems. Subsequently, in this study we provide the integration of hardware simulator with software application in order to obtain a complex dynamic system.

3. Stewart-Gough platform with 3-DOF, architecture and mathematical foundation

Based on the fact that for the construction of real flight simulators with full movement contemplated as cyber-physical systems which include hardware and software, the classic platforms of Stewart Gough of 6-DOF have been used. In the current project, a platform of Stewart Gough has been designed modified to be conducted with a 3-DOF in a limited and low cost manner that restricts the superior ability to perform linear movements on the β (x, y, z) plane. Nonetheless, it will maintain its ability to lead any controlled orientation during an airplane flight in order to receive various effects of spatial disorientation to which the pilots are exposed in real life. Among them are the following that have been simulated in this project: (a) illusion of track; (b) approximation illusions; and (c) illusions due to land degradation or fusion. Through these robotic hardware and software platforms, both military and private pilots may be trained allowing them to develop spatial orientation skills in order to avoid potential catastrophic accidents.

In this research, the architecture proposed by Hunt [13] has been applied, which is known as the 3-RPS parallel robot. Such architecture consists of three identical RPS legs, whose lengths are changed with prismatic joints, while the platform moves with 3-DOF, as illustrated in **Figure 1**.

The basic equations used of the Stewart platform with 3-DOF according to [14–18], have been the following:

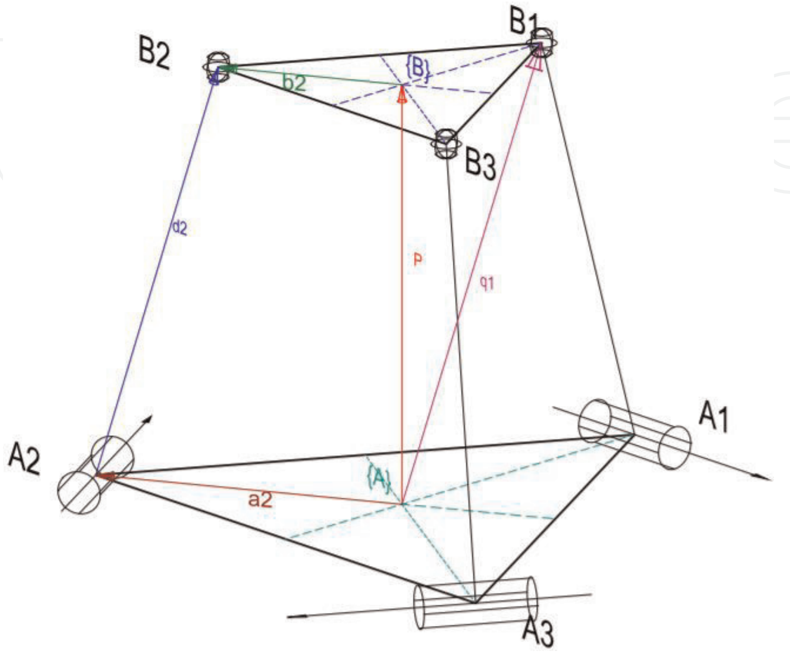

Figure 1.
Schematic illustration of the design of the platform.

Equations of vectors a_1, a_2 and a_3:

$$
\begin{aligned}
a_1 &= [g \;\; 0 \;\; 0]^T \\
a_2 &= \left[-\tfrac{1}{2}g \quad \tfrac{\sqrt{3}}{2}g \quad 0\right]^T \\
a_3 &= \left[-\tfrac{1}{2}g \quad -\tfrac{\sqrt{3}}{2}g \quad 0\right]^T
\end{aligned} \tag{1}
$$

Equations of the vectors b_1, b_2 and b_3:

$$
\begin{aligned}
b_1 &= [h \quad 0 \quad 0]^T \\
b_2 &= \left[-\tfrac{1}{2}h \quad \tfrac{\sqrt{3}}{2}h \quad 0\right]^T \\
b_3 &= \left[-\tfrac{1}{2}h \quad -\tfrac{\sqrt{3}}{2}h \quad 0\right]^T
\end{aligned} \tag{2}
$$

Equations of the prismatic joint represented by three points, namely q_1, q_2 and q_3:

$$q_i = p + {}^{A}R_B \bullet b_i, \; i = 1, 2, 3 \tag{3}$$

Equation of the position of vector p:

$$p = [p_x \quad p_y \quad p_z]^T \tag{4}$$

Equation of the rotation matrix ${}^{A}R_B$.

$${}^{A}R_B = \begin{pmatrix} u_x & v_x & w_x \\ u_y & v_y & w_y \\ u_z & v_z & w_z \end{pmatrix} \tag{5}$$

The equation of the joint q_1 whose procedure is equivalent for q_2 and q_3:

$$q_1 = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} + \begin{bmatrix} u_x & v_x & w_x \\ u_y & v_y & w_y \\ u_z & v_z & w_z \end{bmatrix} \bullet \begin{bmatrix} h \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} p_x + h \bullet u_x \\ p_y + h \bullet u_y \\ p_z + h \bullet u_z \end{bmatrix} \tag{6}$$

4. Flight simulator construction

4.1 Architecture diagram of the flight simulator system

Figure 2 shows the architecture of the system consists of two layers.

4.1.1 First layer

This layer consists of three subsystems that are: (1) subsystem of the flight simulator control, which handles the logic of the virtual flight of the plane applying mathematical, Physics, and aeronautical models; (2) communication subsystem, which manages the connection with the second layer using sockets, the TCP/IP protocol, and flat files containing the information of the roll, pitch and yaw, corresponding to the movement of the plane; and (3) graphical user interface

DOI: http://dx.doi.org/10.5772/intechopen.86586

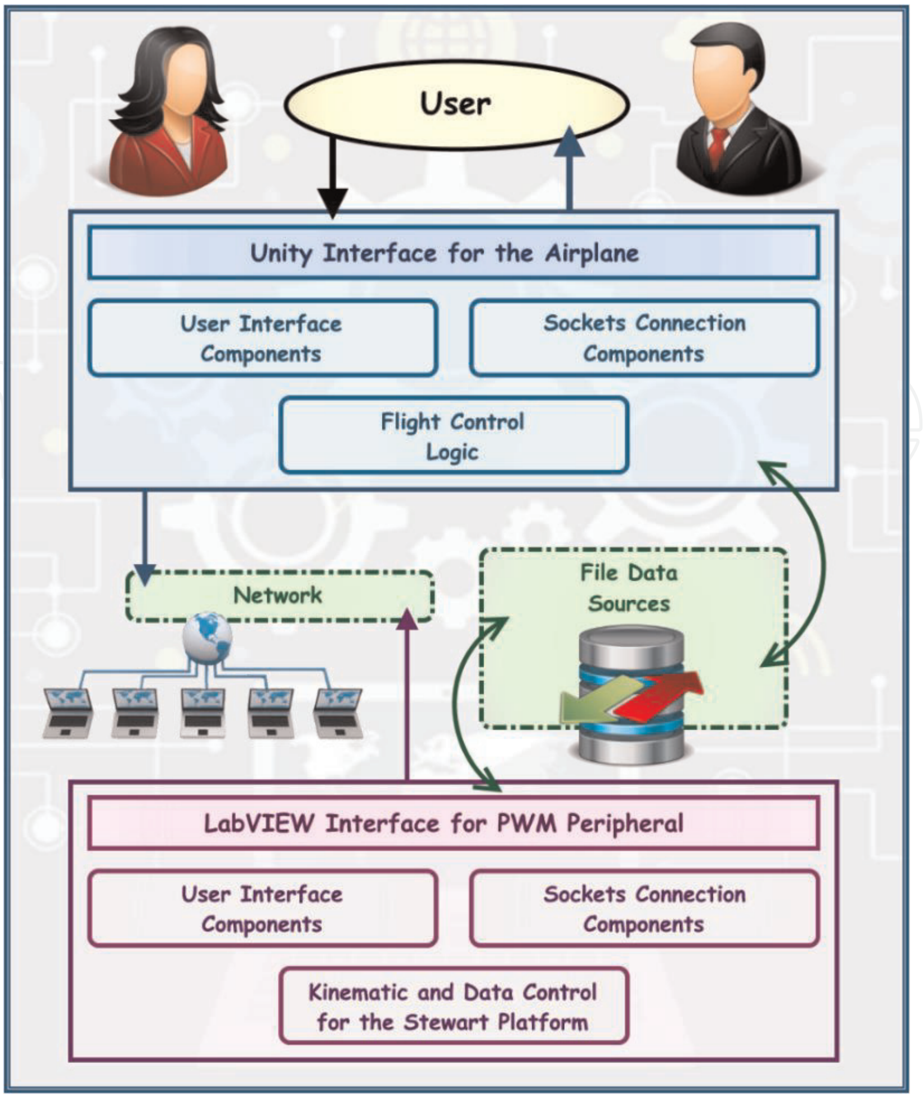

Figure 2.
Architecture diagram of the flight simulator system.

subsystem, which manages the components and assets of Unity 3D to create the virtual world that corresponds to the city of Manta, which is the destination of the plane.

4.1.2 Second layer

This layer consists of three subsystems that are: (1) communication subsystem—manages the connection with the first layer using sockets and reads the flat file with the information of the roll, pitch and yaw of the plane; (2) kinematic control subsystem—obtains the information corresponding to the roll, pitch and yaw of the airplane to be able to reproduce its movements in the Stewart platform with three degrees of freedom using servo motors; and (3) graphical user interface subsystem—manages the LabView components to view in real time the roll, pitch, and yaw values of the aircraft to operate the Stewart platform.

4.2 Flight simulator dynamic system

In this research, we propose a model of the dynamic flight system, which is based on the models proposed by [6, 19, 20]. **Figure 3** shows the interaction between the pilot, the airplane and the Stewart platform with three degrees of freedom. The pilot operates the flight simulator with a joystick, where it controls

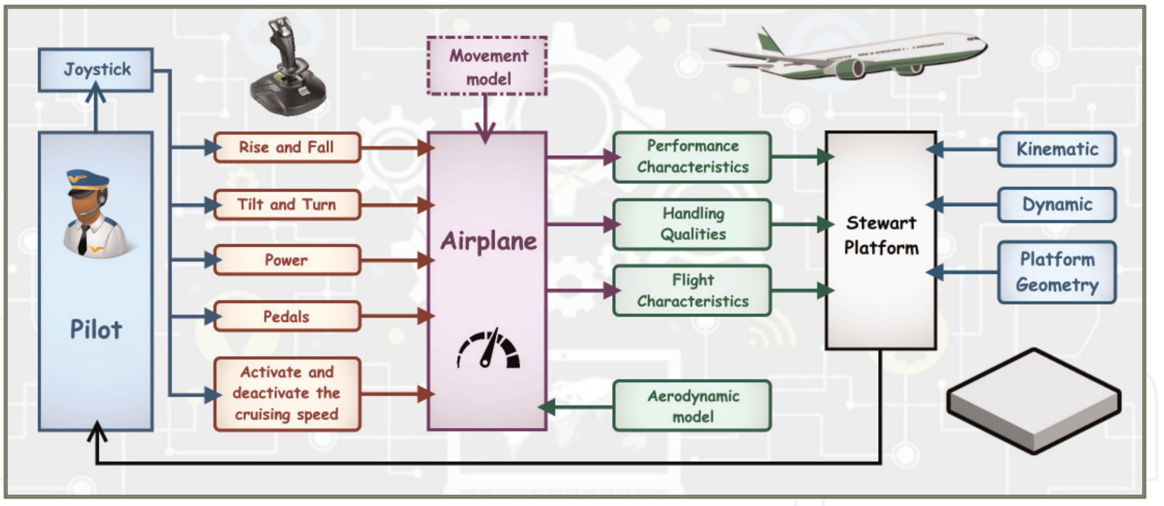

Figure 3.
Proposed flight simulator dynamic system.

the following components: (1) rise and fall of the airplane (range: [−45, 45]); (2) tilt and turn of the airplane (range: [−30 to 30°]); (3) power of the airplane (range: [−1, 1]); (4) pedals of the airplane; and (5) activation and deactivation of the airplane's cruising speed. The Stewart platform reproduces the roll, pitch and yaw movements generated by the flight simulator which is programmed with mathematical models of the flight physics.

For the operation of the Stewart platform with three degrees of freedom, we used a Multifunction DAQ Tested to Data Acquisition Card with pulse-width modulation (PWM) of Texas Instruments brand, which was programmed with LabVIEW to operate the kinematics and dynamics of the system considering the triangular geometry of the platform. This card manages: (1) the performance characteristics, such as control of the power and acceleration of the airplane; (2) handling qualities, such as the control of the speed of advance and rotation of the airplane; and (3) flight characteristics, such as airplane short take-off and landing functions, altitude limit alarm, ground proximity alarm, and landing gear control alarm.

4.3 Design and development of the mechatronic system

The design and construction of the mechatronic product as a complex system was carried out based on Model V, as shown in **Figure 4**. The following tasks have been accomplished according to different areas of knowledge: (1) Mechanical Engineering—This area was applied to adapt and build the mechanical elements of the Stewart platform with three translational degrees of freedom, considering that the proposed dynamic model has no friction and that the kinematic chains are symmetrical and thin; (2) Electrical and Electronic Engineering—This area was applied to control the Stewart platform using servo motors to simplify the inertia and flexibility of the kinematic chains; and (3) Information Technologies—This area was applied to program the data acquisition card with pulse-width modulation (PWM) using LabVIEW and to be able to operate the kinematics and the dynamics of the system with triangular geometry, which reproduces the roll, pitch and yaw movements of the flight simulator.

The code of the control program was based on the block diagrams generated by LabVIEW, for which the data referring to the roll, pitch, and yaw that the flight simulator outputs is read from a flat file. This data is interpreted numerically and processed to be sent to the servo motors that operate the Stewart platform and allow

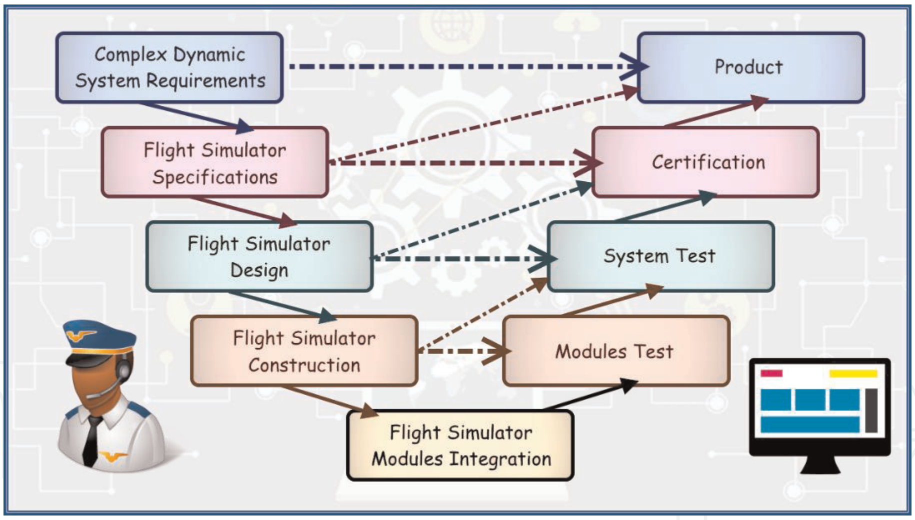

Figure 4.
Application of the V-Model for the flight simulator development.

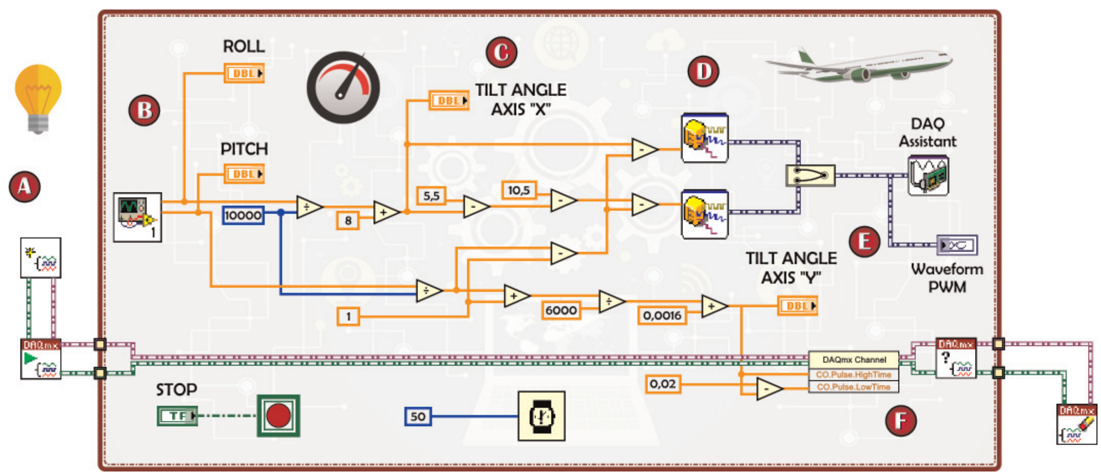

Figure 5.
Block diagram source code for controlling the Stewart platform in LabView.

the movements of the flight simulator to be reproduced. In **Figure 5**, you can see the data flow as described below: (1) programming a pulse-width modulation (PWM) channel; (2) programming functions of downloading and uploading roll, pitch and yaw data that come from the flight simulator; (3) programming conversion of roll, pitch and yaw values into pulse width modulation (PWM) signals to be able to move the servo motors; (4) programming the generation of the square signal by assigning a constant value to the period corresponding to 50 Hz; (5) programming the connection between the processed data with the data acquisition card with pulse-width modulation (PWM) signals to control the servo motors 1 and 3; (6) programming the connection between the processed data with the data acquisition card with pulse-width modulation (PWM) signals to control the servo motor 2.

Figure 6 shows the flow of information from computer 1 (where the flight simulator is installed) to computer 2 (where LabVIEW is installed). The flight simulator saves the data referring to the roll, pitch and yaw in a text file. Then computer 2 accesses this text file through a point-to-point connection using the TCP/IP protocol. Computer 2 receives the data from the text file and operates the control software that works with a data acquisition card with pulse-width modulation (PWM) signals, connected through a USB port to the computer and drives the servo motors of the Stewart platform with three degrees of freedom. Finally, there is a 5 V power supply that powers the servo motors. The mathematical model of the

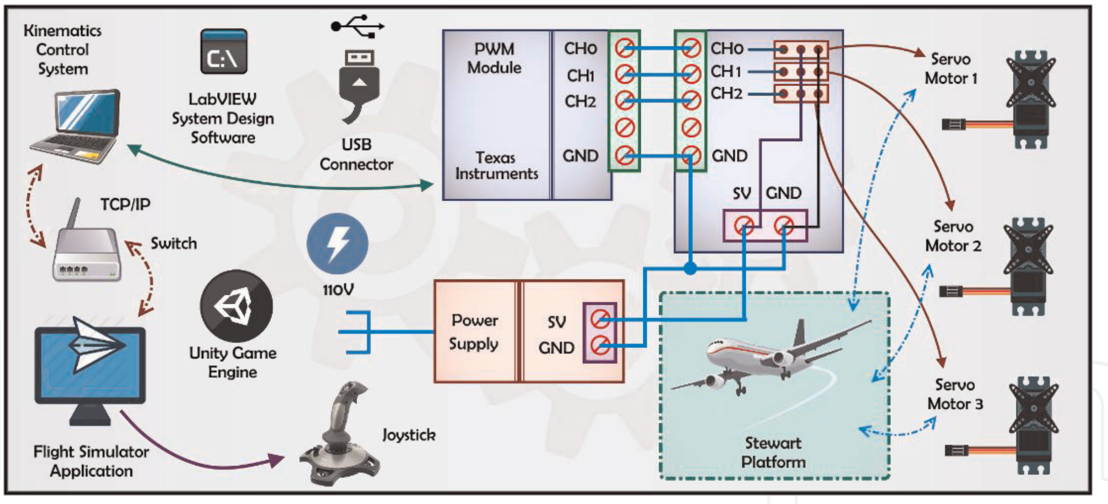

Figure 6.
Infographic of the electric and mechanical diagram of Stewart platform.

flight simulator was programmed with the UNITY 3D framework in computer 1, where the frontal velocity, vertical velocity and lateral velocity are considered allowing the airplane to move in the virtual world as it would be in the real world.

4.4 Mathematical model

In this research, we developed three basic mathematical models for the correct operation of the flight simulator, considering that an aircraft is a mass subjected to different forces such as weight (W), lift (L), thrust (T), and drag (D). In this sense, an analysis of each of these forces and the causes that produce them was made. This allowed the flight simulator to work according to the physical laws that are produced in the air and that the pilot know to properly control of an airplane.

4.4.1 Mathematical model of the front velocity

4.4.1.1 Front velocity

According to [21–23], the thrust force generates the frontal displacement of the airplane. This displacement can be explained using the classical physical laws considering to F_t like a punctual force applied to the mass of the airplane. The following equation is obtained:

$$V_x = V_{ix} + \left(\frac{V_e \cdot k_h}{m_a}\right) \cdot t \tag{7}$$

where F_t is the thrust force; V_e is the engine speed or helix; k_h is the thrust constant; m_a is the mass of the airplane; V_x is the front speed; V_{ix} is the initial velocity of the airplane; and t is the time.

4.4.2 Mathematical model of the vertical velocity

According to [21–23], the lift force generates the vertical displacement of the airplane. This displacement could be explained by using the classical physical laws considering the lift force like a punctual force applied to the mass of the airplane. The following equation is obtained:

$$V_V = V_{vi}\frac{t\cdot\left(\left(\rho\left(V_{ix}+\left(\frac{V_e+k_h}{m_a}\right)\cdot t\right)^2\right)\cdot S_W\cdot C_L\cdot cos(\alpha_t)-W_a\right)}{m_a} \tag{8}$$

where V_v is the vertical velocity of an airplane; V_{vi} is the initial vertical velocity of an airplane; V_{ix} is the initial velocity of the airplane; V_e is the engine velocity of the airplane; k_h is the trust constant; m_a is the mass of an airplane; t is the flight time; S_w is the wing surface; C_L is the coefficient of lift; α_t is the angle of rotation; and W_a is the weight of an airplane.

4.4.3 Mathematical model of lateral velocity

According to [21–23], the resultant force L_y of the sinusoidal component of the lift is responsible for generating the speed of lateral displacement of the aircraft. The following equation is obtained:

$$V_L = V_{iy}\frac{t\cdot\left(\left(\rho\left(V_{ix}+\left(\frac{V_e+k_h}{m_a}\right)\cdot t\right)^2\right)\cdot S_W\cdot C_L\cdot sin(\alpha_t)\right)}{m_a} \tag{9}$$

where V_L is the lateral velocity of an airplane; V_{iy} is the initial velocity on the y-axis of an airplane; V_{ix} is the initial velocity on the x-axis of the airplane; V_e is the engine velocity of an airplane; k_h is the trust constant; m_a is the mass of an airplane; t is the flight time; S_w is the wing surface; C_L is the lift coefficient; and α_t is the angle of rotation.

4.5 Implementation and testing

The CPS complied with a development process for the flight simulator based on the XP cycle that performs iterative and incremental tasks [24, 25]. According to the XP methodology, the work team completed incremental delivery of the software products, based on the following iterations: (1) computational iteration—in this iteration, mathematical models were developed, aerodynamics in physics, management of threads and the design of delegates instance to control the airplane of the virtual world; (2) communication iteration—in this iteration, the processing and transfer of data referring to roll, pitch, and yaw was performed using text files, sockets and a local network; and (3) control iteration—in this iteration, the control system of the Stewart platform was programmed to reproduce the movements of the flight simulator. A multidisciplinary approach supported by the system design engineering allowed the integration of all the modules of the complex dynamic system for its correct operation. **Figure 7** shows the iterations of the CPS based on the XP methodology, applying the model proposed by Drake et al. [26].

Unitary tests were performed on the cyber-physical system that included the computational, communication and control subsystems, both at the software level and at the hardware level. Acceptance tests were also carried out with a group of 40 aircraft pilots, to evaluate the proper functioning of the software application at the end of each iterations. **Figure 8** shows the graphical user interface of the flight simulator developed with the Unity 3D framework, where it is possible to observe the cockpit of a Cessna 172 aircraft operated by a joystick, where the plane is flying over the city of Manta in Ecuador, where the main military aviation schools of the country are located.

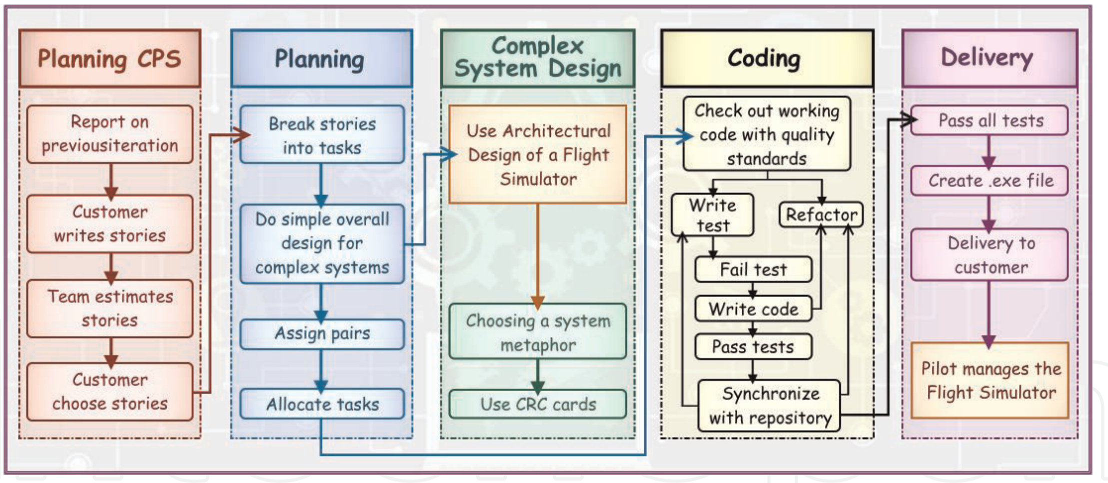

Figure 7.
The Xtreme programming CPS iterations.

Figure 8.
Graphical user interface of the flight simulator.

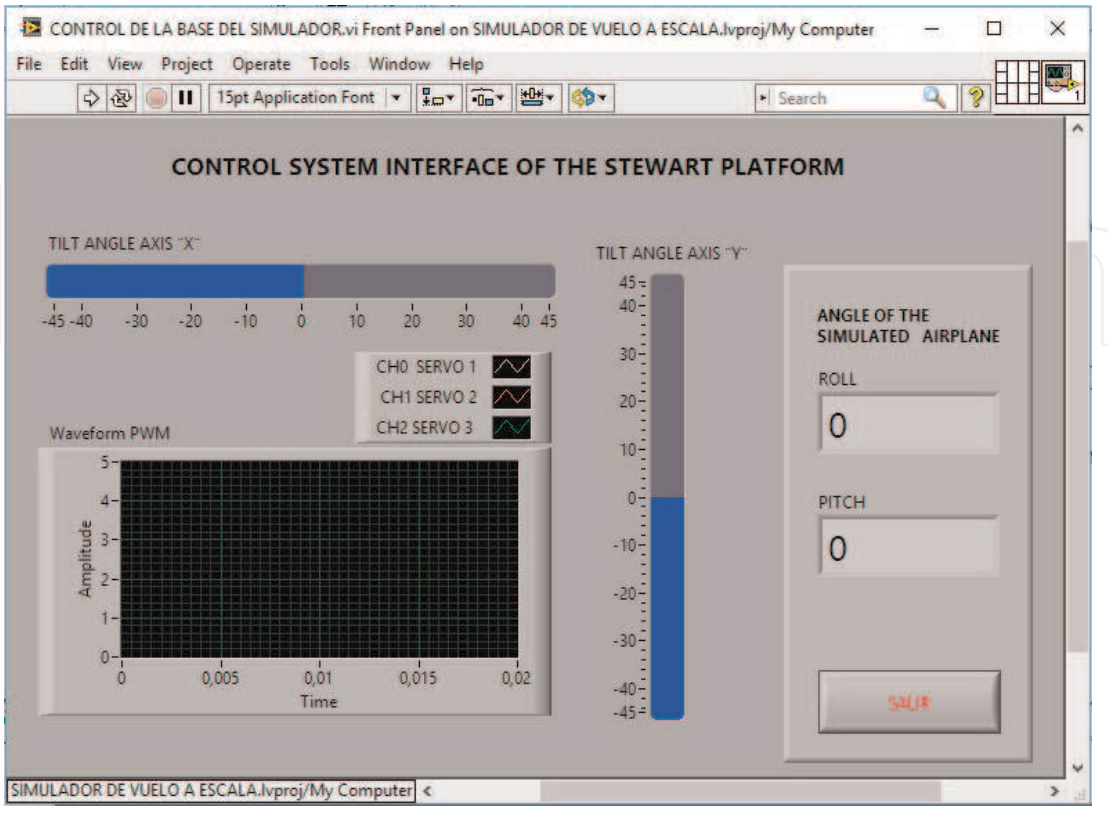

Figure 9.
Control front panel of the Stewart platform.

Figure 9 shows the control front panel of the Stewart platform, where the behavior of the input and output values of the control system developed in LabVIEW can be seen, which reproduces the movements of the flight simulator by integrating a network of two computers that use communication threads with sockets and flat text files, where the information of the three angles of maneuverability of the airplane are found, including: the direction (heading or yaw), elevation (pitch), and angle of bank (roll).

For a proof of concept related to the operation of the flight simulator with the Stewart platform at scale, please access the following links available at: https://youtu.be/pyXP5FlyJYU, https://youtu.be/13d4mFDglmM.

5. Experimental results and discussion

A total population of 40 pilots belonging to the aviation schools of the Armed Forces located in the city of Manta in Ecuador have been chosen to use, test and fly in the developed flight simulator. Of the 40 pilots, 15 correspond to the area of instructors and 25 correspond to the area of students of the aviation schools. After testing the constructed flight simulator, we proceeded to execute the statistical processing, for which we compared six basic characteristics of a flight simulator such as: (a) maneuverability capabilities; (b) motion detection (roll, pitch, yaw); (c) change of plane on the stage; (d) aerodynamic performance; (e) interaction with the virtual simulator; (f) control of the pilot board. These characteristics have been compared with the Microsoft Flight Simulator commercial simulator, obtaining the results indicated in **Figure 10**.

Figure 10, documents the results obtained from a sample of 40 pilots, as indicated above. The reference point has been executed in such a way that a pilot has been tested only in the environment of the flight simulator, being saved of expressing any comment of the experience of testing the simulator with the companions of the aviation schools. Consequently, each evaluation performed by each instructor or apprentice pilot has been free of mutual influence, which leads that the collected data have been an adequate indicator of the perception of the pilots of the flight simulator as a perceptive experience.

In this version of the flight simulator, the lowest value in the characteristics of the simulator corresponds to the control of the pilot's board, since the design and development of three of the six basic flight instruments is almost complete: (a) altimeter; (b) airspeed indicator; (c) vertical speed indicator; (d) attitude indicator; (e) heading indicator; (f) turn indicator. Actually, the simulator is 90% ready and

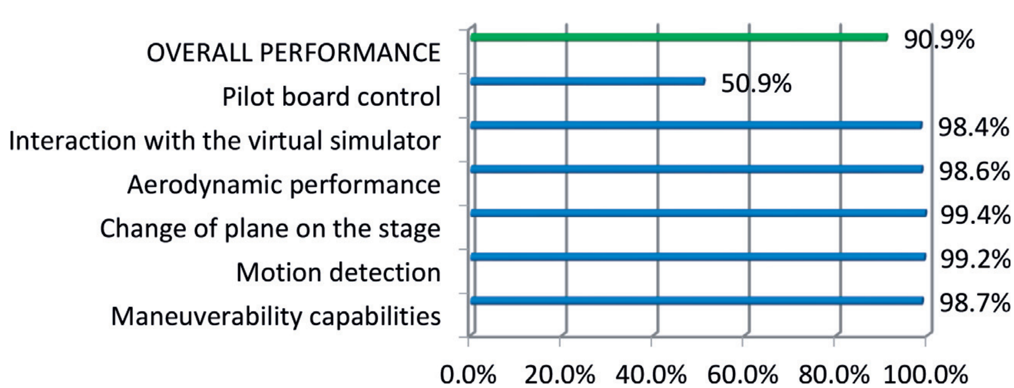

Figure 10.
Average scores of the characteristics of the selected reference points applied for the tests of the flight simulator pilots.

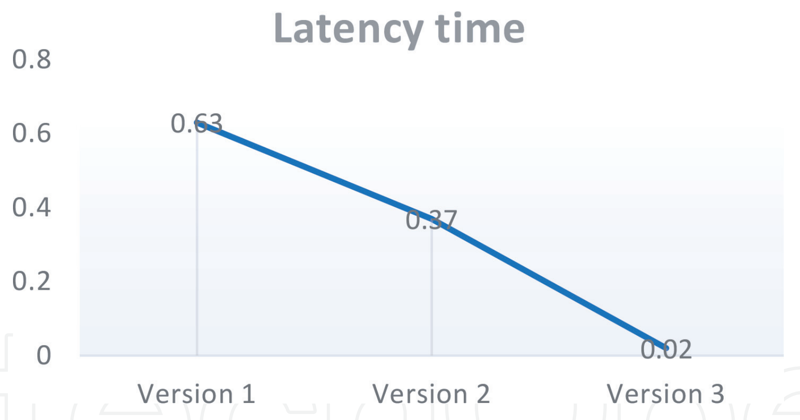

Figure 11.
Mean scores for latency time for all three versions.

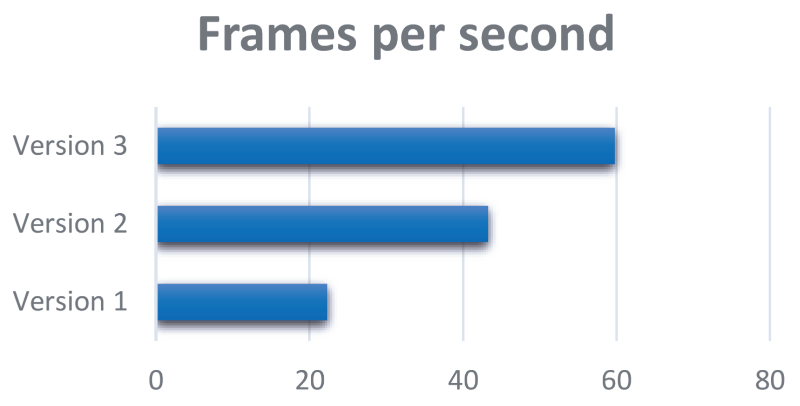

Figure 12.
Mean scores for frames per second for all three versions.

works with a joystick, while it is expected to complete the research project with a rudder, pedals and the power lever connected to the simulator through USB ports.

In addition, three experiments have been performed integrating the algorithms of the mathematical models developed with the Unity 3D game engine with the control algorithms of the Stewart platform of three degrees of freedom developed in LabVIEW, obtaining a communication algorithm that receives the variables of transformation of axes by rotation of the point P (w, x, y, z). Therefore, the Stewart's platform has been able to reproduce the movements of the simulator software, obtains the roll, pitch and yaw values. The evolution of the communication algorithm to reproduce the movement of the Stewart platform in real time with the movements that are generated in the simulator software has been as follows (see **Figures 11** and **12**):

5.1 Version 1

This algorithm contains the movement control variables that were initially written separately and then saved in a text file, in such a way that a latency time of 0.63 s was obtained with 22.3 frames per second.

5.2 Version 2

This algorithm contains the movement control variables that can be written using the Update () function, in such a way that a latency time of 0.37 s was obtained with 43.28 frames per second.

5.3 Version 3

This algorithm contains the movement control variables that can be sent by threads with the Update () function, in such a way that a latency time of 0.02 s was obtained with 59.8 frames per second.

The results obtained from the different versions of the flight simulator are shown in **Figures 11** and **12**.

Based on these obtained results, we have been able to document that the latency time has been reduced from 0.63 to 0.02 s, thus achieving data transmission of almost 60 frames per second. This allowed the reproduction in real time of the movements of the Stewart platform with the movements of the flight simulator software.

6. Conclusions and future work

The objective of this study was to design, develop and implement a flight simulator as a Cyber-Physical system with a Stewart platform at scale with three degrees of freedom. It has been fulfilled principles of System Design Engineering such as the V-Model for Mechatronics and Cyber-Physical Systems; and Software Engineering techniques such as the XP method for the process of design and development of the computer system in terms of communicational, computational and control modules, ensuring the quality of the software. In addition, mathematical models were applied for the calculations of the frontal, vertical and lateral velocities. These formulas were developed and then implemented with the Unity 3D game engine for the Cessna 172 aircraft, in a network of two computers that communicated with each other through flat files consumed by the LabVIEW libraries that allowed reproducing the movements of the flight simulator software application on the Stewart platform scale. This simulator has been created to improve skills and abilities of flight and space disorientation in the training of military pilots of war and combat aircraft. The validation of the proposed solution was made at instructor and student level with several pilots of the aviation schools of the Armed Forces of Ecuador, who have previously been trained in the handling of flight simulation software known as the Microsoft Flight Simulator. The results indicated that this flight simulator supports the development of aircraft control skills and abilities, leading to an increase in its maneuverability and flight capabilities.

As future works, we plan to raise the development and implementation of flight simulators and low-cost spatial disorientation that allow reproducing the movements of the simulation software on a Stewart platform with four degrees of freedom composed of three pistons, electro-valves, microprocessors and other electromechanical elements to reproduce the movement of the roll, pitch, and yaw, as well as to generate at least two or three gravities through the left and right rotational movement of the Stewart platform to disorient the pilot.

Acknowledgements

This study has been partially funded by the Universidad de las Fuerzas Armadas ESPE in Sangolquí, Ecuador within the research project entitled "Construction of a spatial disorientation simulator for contribute to the aviation safety and training of the pilots of Army Forces FF-AA."

Conflict of interest

The authors declare no have conflict of interest.

Author details

César Villacís[1,2], Walter Fuertes[1,2*], Luis Escobar[1,2], Fabián Romero[1,2] and Santiago Chamorro[1,2]

1 Computer Sciences Department, University of the Army Forces-ESPE, Sangolquí, Ecuador

2 Mechanical and Energy Department, University of the Army Forces-ESPE, Sangolquí, Ecuador

*Address all correspondence to: wmfuertes@espe.edu.ec

References

[1] Guerineau B, Bricogne M, Durupt A, Rivest L. Mechatronics vs. cyber physical systems: Towards a conceptual framework for a suitable design methodology. In: 2016 11th France-Japan & 9th Europe-Asia Congress on Mechatronics (MECATRONICS)/17th International Conference on Research and Education in Mechatronics (REM); Compiègne, France; 2016, pp. 314-320

[2] Valis D, Vintr Z, Soares C, Zio E. Dependability of mechatronics systems in military vehicle design. Safety and Reliability for Managing Risk–Proceedings of the European Safety and Reliability Conference 2006 (ESREL). 2006

[3] Mezyk A, Switonski E, Kciuk S, Klein W. Modelling and investigation of dynamic parameters of tracked vehicles. Mechanics and Mechanical Engineering. 2012;**15**(4):115-130

[4] Rizzoni G, et al. Modeling, simulation, and concept design for hybrid-electric medium-size military trucks Defense and Security. Orlando, FL; 2005. pp. 1-12

[5] Escobar L, et al. Design and implementation of complex systems using mechatronics and cyber-physical systems approaches. In: 2017 IEEE International Conference on Mechatronics and Automation (ICMA); Takamatsu, Japan; 2017. pp. 147-154. DOI: 10.1109/ICMA.2017.8015804

[6] Plateaux R, Penas O, Choley J, Mhenni F, Hammadi M, Louni F. Evolution from mechatronics to cyber physical systems: An educational point of view. In: 2016 11th France-Japan & 9th Europe-Asia Congress on Mechatronics (MECATRONICS)/17th International Conference on Research and Education in Mechatronics (REM); Compiegne; 2016, pp. 360-366. DOI: 10.1109/MECATRONICS.2016.7547169

[7] Hager G, Wellein G. Introduction to High Performance Computing for Scientists and Engineers. USA: Taylor and Francis Group, LLC, CRC Press; 2010. ISBN 978-1-4398-1192-4

[8] Derler P, Lee EA, Vincentelli AS. Modeling cyber–physical systems. Proceedings of the IEEE. 2012;**100**(1): 13-28. DOI: 10.1109/JPROC.2011.2160929

[9] Rajkumar R, Lee I, Sha L, Stankovic J. Cyber-physical systems: The next computing revolution. In: Proceedings of the 47th Design Automation Conference on—DAC'10; Anaheim, California; 2010. pp. 731

[10] Lee EA. Cyber-physical systems-are computing foundations adequate. Position paper for NSF workshop on cyber-physical systems: Research motivation, techniques and roadmap. Vol. 2. Citeseer, 2006

[11] Heynen M. Cluster Computing: Distributed Computing Architecture. 1st ed. CreateSpace Independent Publishing Platform. ISBN: 978-1541018563. USA; 2016

[12] Vepa R. Flight Dynamics, Simulation, and Control: For Rigid and Flexible Aircraft. 1st ed. First Published, CRC Press; 2014, 696 pages. eBook ISBN: 9780429101724. DOI: https://doi.org/10.1201/b17346

[13] Hunt KH. Structural kinematics of in-parallel-actuated robot-arms. Journal of Mechanisms, Transmissions, and Automation in Design. 1983;**105**(4):705-712. 8 pages. DOI:10.1115/1.3258540

[14] Alexandrov VV et al. Introduction to Control of Dynamic Systems. Ediciones BUAP. 2009. p. 239. ISBN 978-607-487-095-4

[15] Matlalcuatzi E et al. Diseño del simulador dinámico para pilotos como

un sistema biomecatrónico. In: Memorias del XVI Congreso Latinoamericano de Control Automático, CLCA 2014; Octubre 14–17; Cancún, Quintana Roo, México

[16] Rekdalsbakken W. Design and application of a motion platform in three degrees of freedom. In: Proceedings of SIMS 2005, the 46th Conference on Simulation and Modelling; NO-7005 Trondheim: Tapir Academic Press. pp. 269-279

[17] Rekdalsbakken W. Design and application of a motion platform for a high-speed craft simulator. In: ICM 2006, IEEE 3rd International Conference on Mechatronics. DOI: 10.1109/ICMECH.2006.252493

[18] Villacís C, Navarrete M, Rodríguez I, Romero F, Escobar L, Fuertes W, et al. Real-time flight simulator construction with a network for training pilots using mechatronics and cyber-physical system approaches. 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), Chennai, India; 2017, pp. 238-247. DOI: 10.1109/ICPCSI.2017.8392169

[19] Zheng S, He J, Jin J, Han J. DDS based high fidelity flight simulator. In: WASE International Conference on Information Engineering; IEEE; 2009. DOI 10.1109/ICEI.2009.61

[20] Xinren W, Rongzhen J, Xiaoyuan P, Qin F. Flight Real-Time Simulation System and Technology. Beijing: Press of Beihang University; 2003

[21] Merk RJ, Roessingh JJM. Assessing behaviour of cognitive agents in a flight simulator with fighter pilots. In: 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC); IEEE; 2016

[22] Boril J, Leuchter J, Smrz V, Blasch E. Aviation simulation training in the Czech air force. In: 2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC); IEEE; September 2015. pp. 9A2-1

[23] Villacís C et al. Mathematical models applied in the design of a flight simulator for military training. In: Rocha Á, Guarda T, editors. Developments and Advances in Defense and Security. In: MICRADS 2018. Smart Innovation, Systems and Technologies. vol. 94. Cham: Springer; 2018, pp. 43-57

[24] Beck K. Extreme Programming Explained: Embrace Change. 1st ed. Addison-Wesley Professional. USA: Pearson Education Inc. Nov 17th, 2004, 189 pages. ISBN: 0321-27865-8

[25] Pressman RS. Ingeniería de Software. Un Enfoque práctico. Sexta edición ed. México DF, México: Editorial McGraw-Hill; 2010. ISBN: 9701054733

[26] Drake P, Kerr N, Arbor A. Developing a computer strategy game in an undergraduate course in software development using extreme programming. In: Eighth Annual North-Western Regional Conference; 2006