Científica, vol. 29, núm. 1, pp. 01-22, enero-junio 2025, ISSN 2594-2921, Instituto Politécnico Nacional, MÉXICO
e290109 | DOI: 10.46842/ipn.cien.v29n1a09

ARTICLE

WebVR Platform for Simulation and Control of Electromechanical Systems in a Virtual Environment

Plataforma WebVR para la simulación y el control de sistemas electromecánicos en un entorno virtual

Michel Alejandro **Cruz Martínez**[1], Alejandro **Rodríguez Molina**[2], Gerardo **Hernández Hernández**[3], Mario **Aldape Pérez**[4], Miguel Gabriel **Villarreal Cervantes**[5], Allan Balam **Rueda Gutiérrez**[6]

Tecnológico Nacional de México, Instituto Tecnológico de Tlalnepantla, Estado de México, MÉXICO
[1]ORCID: 0009-0007-4924-2261 | M15250271@tlalnepantla.tecnm.mx
Universidad Autónoma de la Ciudad de México, Colegio de Ciencia y Tecnología, Ciudad de México, MÉXICO
[2]ORCID: 0000-0002-6901-3833 | alejandro.rodriguez.molina@uacm.edu.mx
[3]ORCID: 0009-0001-2268-1930 | gerardo.hernandez.hernandez@uacm.edu.mx
Instituto Politécnico Nacional, Centro de Innovación y Desarrollo Tecnológico en Cómputo, Ciudad de México, MÉXICO
[4]ORCID: 0000-0002-1504-4714 | maldape@ipn.mx
[5]ORCID: 0000-0002-7565-8128 | mvillarrealc@ipn.mx
[6]ORCID: 0009-0002-1887-8148 | allan.rg@tlalnepantla.tecnm.mx

Recibido 16/04/2025, aceptado 25/06/2025.

Abstract
Electromechanical systems play a crucial role in many modern applications. To exploit them effectively, it is essential to tune their controllers appropriately. This tuning usually involves an iterative process of experimenting with the real system on a physical platform. However, to minimize risks and reduce costs, it is more convenient to use simulations of both the systems and their controllers on virtual platforms. Current virtual platforms allow simulations of varying complexity to be carried out using different technologies. Despite this, they have important limitations such as low accessibility, lack of interactive visualization, scarcity of platforms offering accurate simulations of electromechanical systems with kinematic and dynamic models, and dependence on proprietary software and expensive hardware. In addition, emerging technologies such as WebVR, which could increase accessibility and improve immersion at a reasonable cost, are not taken advantage of. To overcome these barriers, this paper proposes a methodology for the development of virtual platforms of electromechanical systems using WebVR. This methodology allows realistic and real-time simulations, with 3D visualizations, interactivity and parameter configuration. The methodology has been successfully applied in the development of a virtual platform for three fully-actuated electromechanical systems: the simple pendulum, the linear inverted pendulum, and the double pendulum.
Index words: virtual platform, WebVR, electromechanical systems, simulation, control.

Resumen
Los sistemas electromecánicos desempeñan un papel crucial en muchas aplicaciones modernas. Para utilizarlos de manera efectiva, es esencial ajustar correctamente sus controladores. Este ajuste generalmente implica un proceso iterativo de experimentación con el sistema real en una plataforma física. Sin embargo, para minimizar riesgos y reducir costos, es más conveniente utilizar simulaciones tanto de los sistemas como de sus controladores en plataformas virtuales. Las plataformas virtuales actuales permiten realizar simulaciones de diversa complejidad mediante diferentes tecnologías. A pesar de esto, presentan limitaciones significativas, como baja accesibilidad, falta de visualización interactiva, escasez de plataformas que ofrezcan simulaciones precisas de sistemas electromecánicos con modelos cinemáticos y dinámicos, y dependencia de software propietario y hardware costoso. Además, tecnologías emergentes como WebVR, que podrían mejorar la accesibilidad y la inmersión a un costo razonable, no están siendo aprovechadas. Para superar estas barreras, este trabajo propone una metodología para el desarrollo de plataformas virtuales para sistemas electromecánicos utilizando WebVR. Esta metodología permite realizar simulaciones realistas en tiempo real con visualización 3D, interactividad y configuración de parámetros. La metodología se ha aplicado con éxito en el desarrollo de una plataforma virtual para tres sistemas electromecánicos completamente actuados: el péndulo simple, el péndulo lineal invertido y el péndulo doble.
Palabras clave: plataforma virtual, WebVR, sistemas electromecánicos, simulación, control.

I. INTRODUCTION

Electromechanical systems consist of mechanical and electrical components that work together to perform specific tasks. These systems are widely used in various areas, such as medicine (in medical instruments or diagnostic equipment) [1], engineering (in specialized machines, actuators and controllers for industry) [2] and in various devices for everyday use, such as means of transportation [3] and household appliances [4]. As a particular type of dynamic systems, electromechanical systems have a number of minimal states or variables that allow describing their behavior at any time [5].

For an electromechanical system to function correctly and effectively fulfill the task for which it was designed, it is necessary for its states to be managed by a control system or controller [6]. Control engineering is responsible for the design, adjustment and optimization of these controllers, ensuring that the system as a whole operates efficiently and accurately. This discipline is essential to ensure stability, dynamic response and performance in a wide range of industrial and technological applications.

Controller tuning involves tuning the parameters or gains of a controller using a model that roughly represents the dynamics of the plant, i.e., the system to be controlled, over time [7]. This tuning can be done empirically, based on direct experience and observation, or automatically, using computational technology to optimize the parameters more accurately and efficiently [8].

Regardless of the approach used to tune the controller of an electromechanical system, this process usually requires multiple experimental iterations on physical platforms. That is, several tests are performed where the performance of the controlled system is evaluated under different configurations of the controller parameters until the specific requirements of the application are met. If these iterations are carried out using the real system, costs (such as operational and maintenance costs) and risks (such as those associated with unexpected or undesired system behavior under certain controller configurations) tend to increase considerably.

For this reason, in most cases, it is preferable to use a simulation of the controlled system in a simulated environment or virtual platform during the iterative process of controller tuning [9]. Simulating the system in a virtual platform involves using models that accurately replicate the operation of real systems, which allows testing and experimentation to understand and analyze the behavior of systems under different environmental and operating conditions, without incurring the costs and risks associated with direct experimentation on real systems. Once the process is completed, the fine-tuned controllers can be implemented in real systems.

Simulations performed on virtual platforms can vary in their level of complexity, ranging from simple graphical representations or basic numerical results to advanced options that employ interactive 3D graphics. When these more advanced simulations include computer-generated 3D graphics, they are known as virtual reality simulations [10]. Virtual reality is a rapidly growing field within engineering, with applications in research and in the development of new technologies [11]. This type of simulations offers a more immersive and detailed interaction with the simulated models, which facilitates the understanding, use and optimization of the systems, allowing to extrapolate their behavior to the real world.

Nowadays, there are several virtual platforms designed to develop simulations of dynamic systems for different purposes. Below are some examples of these platforms, many of them focused on electromechanical systems.

In response to the COVID-19 pandemic, several virtual platforms have been developed for the educational sector as alternatives to experimentation in physical facilities, offering students the possibility to continue their training in simulated environments. A notable example is the work presented in [12], which describes a virtual fluid mechanics platform designed for student instruction. This platform uses the VLAB simulator to generate 2D simulations, employing mathematical models of fluid mechanics controlled by valves and other mechanical components. Another approach in virtual education is found in [13], where a virtual platform for teaching Fourier series in mathematics is presented. This platform allows manipulating electrical and physical signals using Java-based interactive free software. The simulations employ mathematical models to adjust the amplitude and frequency of these signals. In the field of optics and photonics, [14] introduces a virtual platform aimed at teaching these subjects to engineering students, where learning is evaluated by means of 2D optical diffraction and photonic simulations using VLAB.

On the other hand, [15] describes a virtual platform designed for engineering students that allows the implementation of virtual industrial control and automation plants. This platform incorporates electromechanical systems such as pistons and sensors, and uses Autodesk Inventor to develop the 3D environment of the plant, together with MATLAB's Simulink to control the 2D and 3D motions. The control system is implemented using a Programmable Logic Controller (PLC). Similarly, [16] presents a virtual platform to simulate a smart house, integrating mechanical, electronic and electrical systems. The simulation is performed using LabView and a house simulator called HOME I/O, which allows mathematically modeling the house systems, including motion sensors and actuators. A comparable approach is found in [17], where a virtual platform for teaching electrical machines is introduced. This platform allows students to perform practices of different difficulty levels according to their skills, using Web programming for implementation. Although algorithms and flowcharts are handled, electrical models and control elements are not detailed. Finally, the work in [18] explores a virtual platform for teaching chemical reactions, using software such as kineticsTM and MoDSTM to simulate 2D experiments.

The research in [19] presents a non-immersive virtual platform to simulate the behavior of a Mitsubishi Melfa RV 2SDB robotic arm. This simulation, implemented with Unity 3D, integrates mathematical models that allow replicating the control, mechanical and electronic components of the robotic arm. In a different context, [20] develops a remotely controlled power electronics experiment through a virtual platform. This platform includes oscilloscopes and adjustable power electronic circuits, which are operated remotely via Web programming. Although physical devices are used, no electronic models or equations are simulated, and the platform is limited to the operation of the measurement devices. In addition, [21] introduces a virtual platform that simulates the operation of various electrical machines using technologies such as Oculus Rift, Unity3D, and Leap Motion. This platform combines electrical models with control systems to simulate different electromechanical systems. In the field of motion platform simulation, [22] develops a visual compensation scheme for the Stewart 6DOF platform, used in flight simulators. The simulation is carried out with Simulink, while a C++ program is used to adjust the virtual information and apply compensation algorithms. The work of [23] focuses on virtual training with augmented reality to improve motor skills in medical rehabilitation, using virtual reality systems such as HTC VIVE and Oculus Rift. Finally, in [24], a virtual platform is described that simulates the interaction of a patient with a respiratory ventilator, using an electromechanical lung simulator. This simulator incorporates motors, controllers and sensors, in addition to virtualization technology to emulate the normal respiratory functions of a patient.

A different approach is presented in [25], where a 3D virtual simulator for hysteroscopy is evaluated using Leap Motion and Unity 3D. This simulator combines virtual reality with a 3D printed hysteroscope. In the field of industrial simulation, [26] introduces a virtual forklift simulator, developed with Unreal Engine. This simulator includes 3D models manipulated using a Logitech steering wheel and joystick. The paper in [27] describes a 3D simulator to analyze the behavior of aerial manipulator robots in control tasks, using Unity 3D and MATLAB. The simulations are based on kinematic equations and a control law, and integrate a haptic device to operate the robot. On the other hand, [28] presents an interface between a Siemens PLC simulator and a program in Visual Studio, which connects the simulation of an electromechanical plant. This simulation includes interactive elements and 2D models of the plant. In [29], a virtual platform for engineering education using industrial robots and automation systems is developed. The simulation is implemented with MATLAB and Unity3D, employing kinematic and dynamic models for industrial robots, with control elements visualized in 3D. The work in [30] introduces a virtual platform for engineering electrical materials, complemented with an electronic textbook and databases of virtual experiments, implemented with JSON-based Web technology, ASP.Net and Adobe Flash. Finally, [31] describes a virtual platform for testing electrical induction machines, using circuit models and electrical equations. The control is performed by PLC, and the simulation is carried out in a 3D virtual environment.

Based on the previous review, many of the simulations in current virtual platforms present significant limitations that affect both their effectiveness and accessibility. On the one hand, there are simulations that lack a visual interface, showing only numerical results, which can make it difficult to understand the concepts and limit interaction with users. In addition, there is a notable shortage of platforms dedicated to electromechanical systems, and those that do include them often do not use kinematic or dynamic models, which decreases the fidelity of the simulations compared to reality. This lack of accuracy is compounded by the absence of control elements, which restricts the development of advanced tasks such as controller tuning.

Another important limitation is that most simulators require the installation of specialized software, which is often proprietary, which restricts their accessibility and may involve additional costs. In addition, the need to acquire expensive hardware for their operation can be a considerable obstacle, especially in contexts where resources are limited. Finally, it has been observed that current virtual platforms have not taken advantage of emerging technologies such as Web Virtual Reality (WebVR), which could offer more accessible, immersive and flexible solutions for the development and exploitation of simulations of these systems and their environments.

Aiming to overcome these limitations, this work proposes a methodology to develop virtual platforms for electromechanical systems, taking advantage of the benefits of WebVR technology, such as its low cost and high accessibility. It is intended that the platforms developed under this methodology simulate kinematic, dynamic and control aspects of electromechanical systems, providing realistic 3D visualizations of their behavior in real time. In addition, we seek to integrate interactive mechanisms that facilitate the manipulation of the visualizations and the configuration of various parameters of the simulation.

The remainder of this paper is organized as follows. Section 2 briefly describes the stages of the proposed methodology. In Section 3, the methodology is applied in the development of a virtual platform for three fully-actuated electromechanical systems: the simple pendulum, the linear inverted pendulum, and the double pendulum. Finally, Section 4 presents conclusions and offers future perspectives on this work.

II. Development methodology

The stages that make up the methodology for the development of virtual platforms in WebVR, intended for the simulation and control of electromechanical systems, are illustrated in Fig. 1.

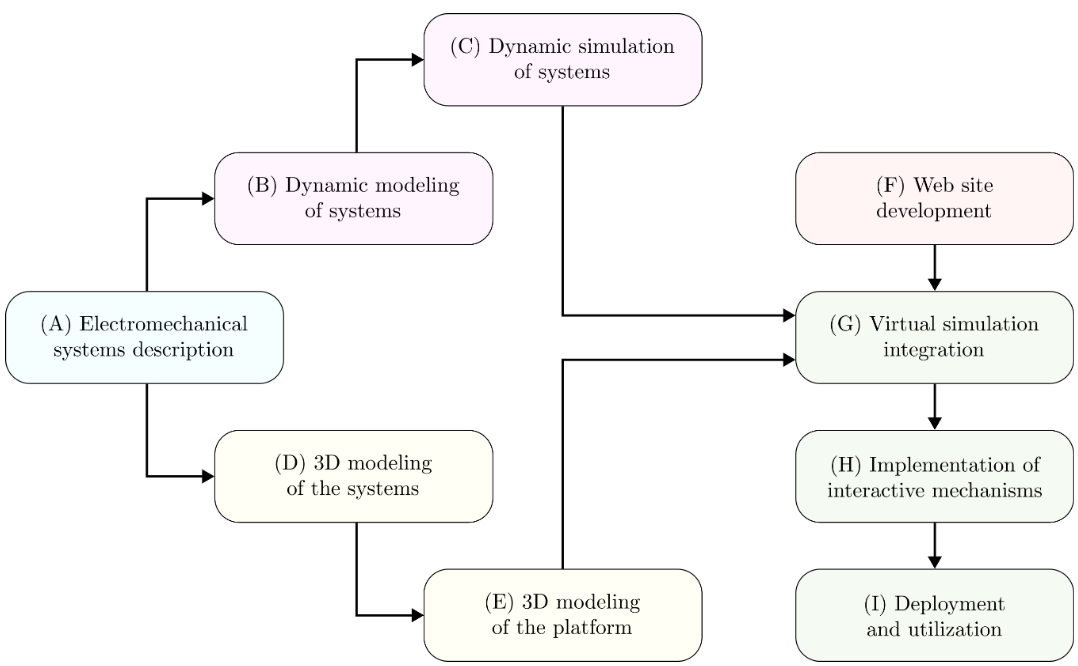


Fig. 1. Methodology for the development of virtual platforms in WebVR for the simulation and control of electromechanical systems.

A. Electromechanical systems description

The first stage focuses on the analysis of the electromechanical systems that will integrate the virtual platform. This analysis allows identifying and describing the most relevant aspects of each system and its operation, which is essential for its subsequent modeling or abstraction in the different stages of the methodology. Key information to be analyzed for any electromechanical system includes its dimensions in a standard measurement system, the movements that its mechanical parts can perform (both independently and together), the electrical inputs that generate those movements and their limitations, the states that describe its behavior at any time, the parameters (internal or from the operating environment) involved in its operation, the initial configuration in terms of states, the controllers, and the operational constraints. In addition, it is crucial to collect information about the physical appearance of the systems and their environment (i.e., the platform and other objects that might be present), in order to create models or visual representations that are familiar to users.

B. Dynamic modeling of systems

Due to the very nature of electromechanical systems, there are well-defined electrical and mechanical rules that allow us to derive accurate models of their behavior at different levels of detail. At this stage, these rules are used to obtain dynamic models of the systems that make up the platform. A dynamic model is a mathematical representation that describes how the relationships between the inputs and outputs of a dynamic system vary over time. Like any model, the accuracy with which the dynamic model reflects the real system depends on the values of its parameters. Some of these values can be derived from the system description performed in step (A), while others require estimation through a system identification process based on data or measurements of the real system [32].

C. Dynamic simulation of systems

Dynamic simulation consists of using the dynamic models obtained in step (B) to predict, by means of a computer, the time evolution of the states of the electromechanical systems from their initial configurations, taking into account the operational constraints, both defined in step (A). Depending on the characteristics of the model, its use can be direct or indirect. In the indirect case, in addition to the model, some numerical or computational method is required to make the predictions. It is advisable to encapsulate the functionalities of the dynamic simulation in a computer program built with Web technology, particularly with JavaScript (JS), to facilitate its use in other stages.

As a result of the dynamic simulation, values of the states of the systems are obtained in discrete sampling intervals with a reasonably small size. These values will be used in later stages to manipulate visual representations of the systems in real time, creating a realistic perception of their motion. Therefore, it is crucial to consider that if the sampling intervals are too large, the visualization of the systems will not be smooth. On the other hand, if the intervals are too small, the computational load will increase proportionally, which could affect the smoothness of the visualization.

D. 3D modeling of the systems

This type of modeling refers to the creation of visual representations of dynamic systems and other objects present on the platform, using 3D geometries that can be visualized on a computer. This task requires specialized software that allows generating and manipulating 3D geometries through operations such as translation, rotation, scaling and extrusion. In addition, these tools allow the assignment of materials, i.e., properties of the surfaces of 3D objects that determine their appearance, such as color or texture. Although there are many 3D modeling software options on the market, free software alternatives such as Blender [33] are highly recommended to overcome the cost limitations observed in other virtual platforms.

Since the methodology contemplates the deployment of the virtual platform on the Web, it is important to consider certain aspects during modeling to optimize the visualization of the 3D elements. In this regard, care must be taken to ensure that the polygons that define the surfaces of the 3D objects are well distributed and that there is no interference between the independent geometries that make up the objects. The dimensions and appearance of the 3D objects should be congruent with those of the real objects, as defined in step (A). Fine details in the models should be moderated to avoid excessive computational cost when displaying them, and it is recommended that the moving parts of the virtual electromechanical systems be identified with representative names to facilitate their manipulation in later stages.

E. 3D modeling of the platform

In this stage, the 3D models of the electromechanical systems and other objects present on the platform, obtained in stage (D), are arranged within the same virtual space. This arrangement of the models should be similar to that of the real platform, so that it is familiar to the end users. The same 3D modeling software used in stage (D) can be used for this purpose.

Once the 3D elements are organized in the same environment, the platform can be exported to a format that facilitates manipulation on the Web. In particular, this methodology proposes the use of the glTF format (GL Transmission Format), as it allows an optimized transmission and loading of the 3D environments, and facilitates the manipulation of its elements using Web technologies such as JS).

F. Web site development

This refers to the use of Web technology to create a customized Web site or page that will host the virtual platform and allow interaction with it [34]. In this process, the interactive, navigation and content elements in the site's user interface, as well as their structure and layout, are implemented using HTML (HyperText Markup Language). The visual appearance of these elements is defined using CSS (Cascading Style Sheets). The real-time manipulation or update of these elements and the handling of the interactivity related to the triggered events is achieved through JS.

G. Virtual simulation integration

In this stage, the dynamic simulation of the electromechanical systems, described in stage (C), is performed in real time, i.e., aligning the selected sampling interval with the time elapsed in reality. The data obtained are used to manipulate the 3D objects representing these systems within the virtual platform, developed in step (E).

First, the 3D model of the virtual platform in glTF format, obtained in step (E), is loaded into the Web site created in step (F). Then, the virtual environment is configured using JS to obtain a proper visualization of the platform. This configuration includes the adjustment of the elements that simulate the lighting and the camera, i.e., the point of view of the virtual space. This viewpoint is rendered as a visible image within the site and is periodically updated to visually reflect changes in the platform, providing a sense of dynamism to the user. In addition, references are created to the moving elements of the 3D objects on the platform, previously identified with representative names, to facilitate their manipulation.

Secondly, the functionality related to the dynamic simulation of the electromechanical systems, obtained in stage (C), is also loaded into the website in stage (F). Next, variables are created in JS to store the current states of the simulation.

Finally, the integration of the results of stages (C) and (E) is achieved by implementing a scheduled task in JS, which is executed each time a real-time sampling interval elapses. This task is in charge of calculating a step forward in the dynamic simulation of the systems, i.e., obtaining the states of the systems in a future sampling interval. Once this calculation has been performed, the state information is immediately used to manipulate the position and orientation of the moving elements in the 3D virtual models of the electromechanical systems.

H. Implementation of interactive mechanisms

This stage is responsible for receiving and processing all actions or inputs generated by the user on the Web site created in stage (F) to trigger changes in the integrated virtual simulation of stage (G). In this way, functionalities can be added with JS that allow the user to navigate the virtual platform using their computer input devices, enhancing their level of immersion in the environment. In addition, the user can be allowed to configure different aspects of the simulation, such as the tasks to be performed by the electromechanical systems, their initial conditions and other control elements, by processing the input fields in the user interface with JS. Through other interactive elements in the interface, such as buttons, the user can make decisions, such as when to start or stop the simulation, using functions implemented in JS.

I. Deployment and utilization

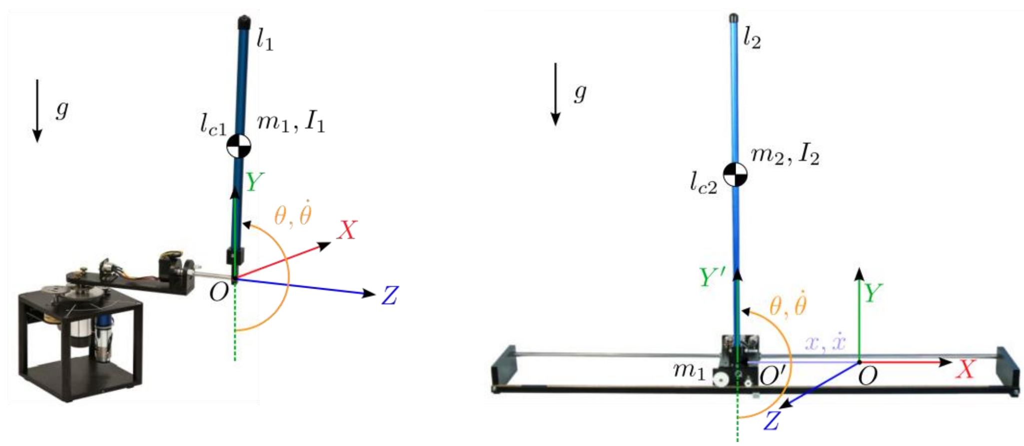
The deployment of the interactive Web site obtained at the end of step (H) for the simulation and control of electromechanical systems on a virtual platform consists of making it accessible to end users by hosting it on a Web server. Since a WebVR application does not require specialized hardware or software infrastructure, nor large amounts of server-side processing, the virtual platform can be effectively deployed on an inexpensive or even free hosting service. Once deployed, end users can access the platform from anywhere, anytime, using only an up-to-date Web browser on any type of computing device, from a low- to mid-range mobile device to a high-capacity workstation. This helps overcome many of the limitations present in today's virtual platforms.

III. Development of a virtual platform for electromechanical systems

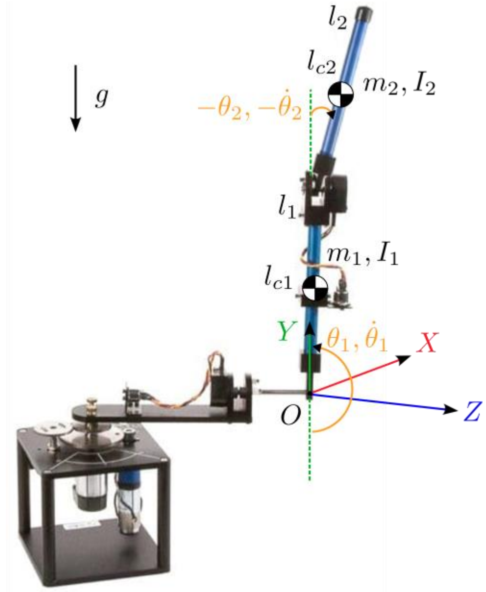
This section analyzes the results obtained during the development of a virtual platform in WebVR for the simulation and control of electromechanical systems, based on the methodology presented in this work. It is assumed that the real platform incorporates two electromechanical systems: a simple pendulum and a linear inverted pendulum, both fully actuated.

A. Electromechanical systems description

For the purposes of this work, systems visually similar to those shown in Fig. 2 are considered. In addition, it is assumed that other objects and office furniture are also found within the virtual platform. All measurements taken during the description stage are in the International System of Units.



(a) Simple pendulum (b) Inverted pendulum



(c) Double pendulum

Fig. 2. Electromechanical systems included in the platform.

The simple fully-actuated pendulum (ASP) depicted in Fig. 2(a) consists of a pendulum of length l_1, whose end is fixed coupled with a base at the origin O, and which can rotate freely around the Z-axis thanks to the torque τ generated by an electric motor. The state describing its behavior at any instant of time t is given by $z = [\theta, \dot{\theta}]^T$, where θ and $\dot{\theta}$ are the angular position and velocity of the pendulum measured with respect to the $-Y$-axis. This system is considered to be influenced by Earth gravity and nonconservative frictional forces, and is initially found to be in stable equilibrium, i.e., $z = [0,0]^T$. In addition, a position regulation task is set for the system with the desired state $z_d = [\theta_d, 0]^T$, where θ_d is the desired angular position for the pendulum. The controller used to perform this task is a Proportional Integral Derivative (PID), as shown in Eq.1 [35], where $u = \tau$ is the control signal calculated from the error $e = \theta_d - \theta$ at any time, using the proportional k_p, integral k_i and derivative k_d gains.

$$u = k_p e + k_i \int e \, dt + k_d \dot{e} \tag{1}$$

On the other hand, the fully-actuated inverted pendulum (AIP) shown in Fig. 2(b) has a moving base that is linearly displaced by the force f. This force is generated by a moving belt driven by an electric motor. The linear motion of the base occurs within the limits defined by $[x_{min}, x_{max}]$ along the X-axis, which belongs to a fixed coordinate system with origin O at the center of the rigid frame of the system. At the origin O', which coincides with the position of the moving base, is located the end of a pendulum of length l_2, which can rotate freely around Z' thanks to the torque τ generated by another electric motor. The state of this system at any instant of time t is defined as $z = [x, \theta, \dot{x}, \dot{\theta}]^T$, where $x, \dot{x}$ represent the linear position and velocity of the moving base with respect to O, and $\theta, \dot{\theta}$ are the angular position and velocity of the pendulum measured from $-Y'$. This system is affected by Earth gravity and frictional forces. Initially, the moving base of the AIP is at the center of the rigid frame of the system and the pendulum is in stable equilibrium, i.e., $z = [0,0,0,0]^T$. As with the ASP, the AIP is sought to perform a position regulation task, with a desired state of $z_d = [x_d, \theta_d, 0,0]^T$, where x_d and θ_d are the desired positions of the moving base and pendulum, respectively. To achieve this, a pair of PID controllers is employed, as shown in Eq. 2, where $u_1 = f$ and $u_2 = \tau$ are the control signals calculated from the errors $e_1 = x_d - x$ and $e_2 = \theta_d - \theta$ at any instant of time, using the proportional k_{p1} and k_{p2}, integral k_{i1} and k_{i2}, and derivative k_{d1} and k_{d2} gains.

$$u_1 = k_{p1}\, e_1 + k_{i1} \int e_1\ dt + k_{d1}\, \dot{e}_1$$
$$u_2 = k_{p2}\, e_2 + k_{i2} \int e_2\ dt + k_{d2}\, \dot{e}_2 \qquad (2)$$

Finally, the fully-actuated double pendulum (ADP) observed in Fig. 2(c) consists of a first pendulum of length l_1 with one end coupled by means of a rotating joint to a fixed base at the origin O, and the other end coupled, also by means of a rotating joint, to the end of a second pendulum of length l_2. Both pendulums can rotate freely around the Z-axis by the motion due to the torques τ_1 and τ_2 generated by two electric motors. The state of the ADP at any instant of time t is defined as $z = \left[\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2\right]^T$, where $\theta_1, \dot{\theta}_1$ are the angular position and velocity of the first pendulum measured from $-Y'$, and $\theta_2, \dot{\theta}_2$ are the angular position and velocity of the second pendulum measured with respect to θ_1. This system is also affected by Earth gravity and frictional forces. Initially, both pendulums of the ADP are in stable equilibrium, i.e., $z = [0,0,0,0]^T$. As in the case of the other electromechanical systems, the ADP aims to perform a position regulation task, considering a desired state of $z_d = [\theta_{1d}, \theta_{2d}, 0,0]^T$, where θ_{1d} and θ_{2d} are the desired angular positions of both pendulums. Two PID controllers are utilized with this purpose. These are in the form of Eq. 2, where $u_1 = \tau_1$ and $u_2 = \tau_2$ are the control actions obtained with the errors $e_1 = \theta_{1d} - \theta_1$ and $e_2 = \theta_{2d} - \theta_2$ at any instant of time, using the proportional k_{p1} and k_{p2}, integral k_{i1} and k_{i2}, and derivative k_{d1} and k_{d2} gains.

B. Dynamic modeling of systems

At this point, the Euler-Lagrange method is used to derive the dynamic models of the ASP, AIP and ADP. The Euler-Lagrange method is a generalization of Newton's laws that allows obtaining the equations of motion of mechanical systems from the analysis of the potential and kinetic energies of their independent components [36].

This method can be carried out by performing the following steps for a wide variety of electromechanical systems:

1. Select the generalized coordinates that allow describing the behavior of the system at any instant of time. Usually, each generalized coordinate q_i is related to the angle θ_i or the displacement x_i of each joint in the system, considering that it has revolution or prismatic motion, respectively.

2. Calculate the total kinetic energy of the system as the sum of the kinetic energy of each rigid body that makes up the system. This is calculated from the mass m_i, the inertia I_i, the velocity of the center of mass v_i and, if applicable, the rotational velocity w_i (in motions of revolution):

$$K = \sum_{i=1}^{n} \left(\frac{1}{2} m_i \|v_i\|^2 + \frac{1}{2} I_i w_i^2 \right) \qquad (3)$$

3. Calculate the potential energy of the system due to gravity as the sum of the potential energy of each rigid body that makes it up. It is obtained from the mass m_i, the acceleration of gravity g, and the vertical component of the position of its center of mass h_i:

$$V = \sum_{i=1}^{n} m_i g h_i \tag{4}$$

4. Define the Rayleigh dissipation function to model non-conservative effects in the system such as frictions. For the term corresponding to each generalized coordinate q_i, the viscous friction coefficient b_i is used:

$$D = \sum_{i=1}^{n} b_i \dot{q}_i^2 \tag{5}$$

5. Form the Lagrangian:

$$L = K - V \tag{6}$$

6. Based on the Lagrangian, calculate the terms of the equation of motion with dissipation corresponding to each generalized coordinate q_i. In each equation of motion, the generalized input Q_i can be either a force F_i or a torque τ_i when the coordinate is related to a prismatic or a revolute joint, respectively:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} + \frac{\partial D}{\partial \dot{q}_i} = Q_i \tag{7}$$

7. Rewrite the equations of motion obtained in matrix form to facilitate their transformation to the state form.

Applying this method, the behavior of the ASP shown in Fig. 2(a) in state space for any instant of time t is described by Eq. 8, where m_1 is the mass of the pendulum, l_{c1} is the distance from the origin O to its center of mass, b_1 is the coefficient of viscous friction due to air friction, I_1 is its inertia, u is the control signal, and g is the gravity acceleration.

$$\dot{z}(z,u,t) = \begin{bmatrix} \dot{\theta} \\ \dfrac{u - m_1 g l_{c1} \sin\theta - b_1\dot{\theta}}{m_1 l_{c1}^2 + I_1} \end{bmatrix} \tag{8}$$

On the other hand, the operation of the AIP depicted in Fig. 2(b) is described in state space by the matrix equation in Eq. 9. In this equation, m_1 and m_2 are the masses of the moving base and the pendulum, respectively; b_1 and b_2 are the viscous friction coefficients associated with the motion of the mechanical components and the air friction; $u = [u_1, u_2]^T$ represents the control input; l_{c2} is the distance to the center of mass of the pendulum from O'; I_2 is its inertia; and g is the gravity acceleration.

$$\dot{z}(z,u,t) = \begin{bmatrix} \dot{q} \\ \widehat{M}^{-1}(u - \hat{C}\dot{q} - \hat{G}) \end{bmatrix} \tag{9}$$

with:

$$q = [x, \theta]^T$$
$$\dot{q} = [\dot{x}, \dot{\theta}]^T \tag{10}$$

$$\hat{M} = \begin{bmatrix} m_1 + m_2 & m_2 l_{c2} \cos\theta \\ m_2 l_c \cos\theta & m_2 l_{c2}^2 + I_2 \end{bmatrix}$$

$$\hat{C} = \begin{bmatrix} b_1 & -m_2 l_c \dot{\theta} \sin\theta \\ 0 & b_2 \end{bmatrix}$$

$$\hat{G} = [0, m_2 g l_{c2} \sin\theta]^T$$

Finally, the dynamic behavior of ADP from Fig. 2(c) is established in state space by the matrix equation in Eq. 11, where m_1 and m_2 are the masses of both pendulums; b_1 and b_2 are their viscous friction coefficients related to the air friction; $u = [u_1, u_2]^T$ is the control input; l_{c1} and l_{c2} are the distance to the center of mass of each pendulum measured from the nearest end to the base; I_1 and I_2 are their moments of inertia; and g is the gravity acceleration.

$$\dot{z}(z,u,t) = \begin{bmatrix} \dot{q} \\ \hat{M}^{-1}(u - \hat{C}\dot{q} - \hat{G}) \end{bmatrix} \quad (11)$$

with:

$$q = [\theta_1, \theta_2]^T$$

$$\dot{q} = [\dot{\theta}_1, \dot{\theta}_2]^T$$

$$\hat{M} = \begin{bmatrix} m_1 l_{c1}^2 + m_2 l_1^2 + m_2 l_{c2}^2 + 2 m_2 l_1 l_{c2} \cos\theta_2 + I_1 + I_2 & m_2 l_{c2}^2 + m_2 l_1 l_{c2} \cos\theta_2 + I_2 \\ m_2 l_{c2}^2 + m_2 l_1 l_{c2} \cos\theta_2 + I_2 & m_2 l_{c2}^2 + I_2 \end{bmatrix} \quad (12)$$

$$\hat{C} = \begin{bmatrix} b_1 - 2 m_2 l_1 l_{c2} \dot{\theta}_2 \sin\theta_2 & -m_2 l_1 l_{c2} \dot{\theta}_2 \sin\theta_2 \\ m_2 l_1 l_{c2} \dot{\theta}_1 \sin\theta_2 & b_2 \end{bmatrix}$$

$$\hat{G} = \begin{bmatrix} m_1 g l_{c1} \sin\theta_1 + m_2 g l_1 \sin\theta_1 + m_2 g l_{c2} \sin(\theta_1 + \theta_2) \\ m_2 g l_{c2} \sin(\theta_1 + \theta_2) \end{bmatrix}$$

The dynamic parameters used in the above models are assumed to be known and are presented in Table 1. Details on how these dynamic models were obtained can be found at [37].

TABLE 1
PARAMETERS USED IN THE ASP AND AIP DYNAMIC MODELS.

Parameter	Value for the ASP	Value for the AIP	Value for the ADP
$\boldsymbol{m_1}$	$1.0000\ (kg)$	$1.0000\ (kg)$	$1.0000\ (kg)$
$\boldsymbol{m_2}$	–	$1.0000\ (kg)$	$1.0000\ (kg)$
$\boldsymbol{l_1}$	$1.0000\ (m)$	–	$1.0000\ (m)$
$\boldsymbol{l_2}$	–	$1.0000\ (m)$	$1.0000\ (m)$
$\boldsymbol{l_{c1}}$	$0.5000\ (m)$	–	$0.5000\ (m)$
$\boldsymbol{l_{c2}}$	–	$0.5000\ (m)$	$0.5000\ (m)$
$\boldsymbol{I_1}$	$0.0050\ (kg\ m^2)$	–	$0.0050\ (kg\ m^2)$
$\boldsymbol{I_2}$	–	$0.0050\ (kg\ m^2)$	$0.0050\ (kg\ m^2)$
$\boldsymbol{b_1}$	$0.5000\ (N\ m\ s)$	$0.5000\ (N\ s/m)$	$0.5000\ (N\ m\ s)$
$\boldsymbol{b_2}$	–	$0.5000\ (N\ m\ s)$	$0.5000\ (N\ m\ s)$
$\boldsymbol{g}$	$9.8100\ (\ m/s^2)$	$9.8100\ (\ m/s^2)$	$9.8100\ (\ m/s^2)$

C. Dynamic simulation of systems

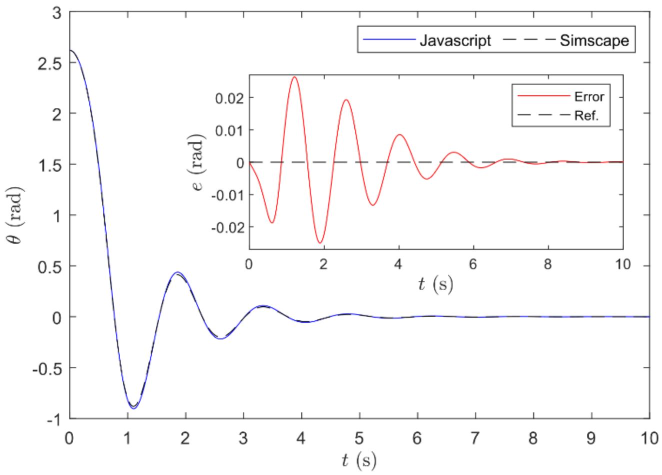
The models of the ASP, AIP, and ADP systems are first order differential equations with initial conditions, whose numerical solution at any time instant t allows predicting their future state at instant $t + dt$, where dt es el intervalo de muestreo. is the sampling interval. Because of this, the dynamic models of the systems cannot be used directly and must go through a numerical integration process. In this work, the Euler numerical integration method, also known as *ode1*, is used due to its computational efficiency and ease of implementation in any development technology, including JS. For this purpose, a sampling interval of $dt = 0.005\ (s)$ is used with the objective of generating smooth simulations and smooth visualizations of the virtual electromechanical systems.

As a result of this stage, we obtain the JS code that encapsulates the computations of each dynamic model and the functionality that allows predicting the future states $z(t + dt)$ of the system, one step dt ahead in time, using $ode1$, from the current state $z(t)$ and the control input $u(t)$.

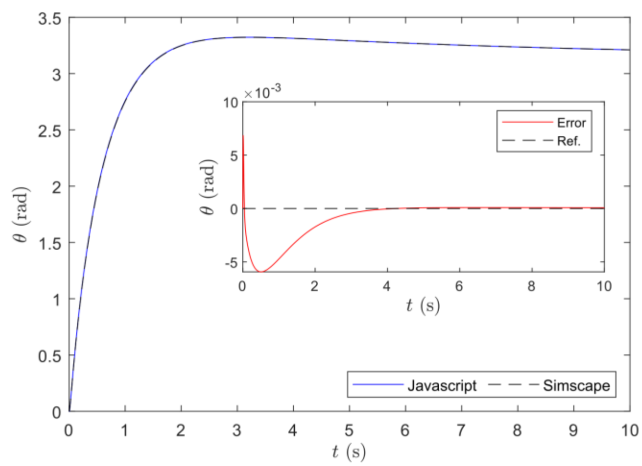
At this point, it is worth verifying that the simulations performed in JS adequately describe the behavior that would be expected in the real world of the electromechanical systems studied. To this end, JS simulations of the three systems are carried out under controlled conditions, and the results are compared with those obtained using Simscape. Simscape is a MATLAB tool that allows to graphically model multidomain physical systems using blocks, and to perform dynamic simulations of these systems. This tool is integrated with Simulink and uses advanced numerical solvers to handle differential equations of complex physical systems. Thanks to this, Simscape allows to obtain simulations with a high degree of accuracy and fidelity with respect to the real behavior of the systems.

For each electromechanical system, simulations are performed considering two controlled scenarios: (1) using initial conditions far from equilibrium without the application of a control signal, in order to observe the free response of the system, and (2) using initial conditions in stable equilibrium and applying control signals for the regulation task. These comparisons allow validating that the JS implementation adequately represents the expected dynamics of the physical system. In this sense, the initial conditions used for the first scenario are: $z_0 = \left[\frac{150}{180}\pi, 0\right]^T$ for the ASP, $z_0 = \left[0, \frac{150}{180}\pi, 0,0\right]^T$ for the AIP, and $z_0 = \left[0, \frac{150}{180}\pi, 0,0\right]^T$ for the ADP. For the second scenario, the gains of the PID controller are: $k_p = 40$, $k_i = 5$, and $k_d = 20$ for the ASP; $k_{p1} = k_{p2} = 40$, $k_{i1} = k_{i2} = 5$, and $k_{d1} = k_{d2} = 20$ for the AIP; and $k_p = 40$, $k_i = 5$, and $k_d = 20$ for the ASP; $k_{p1} = k_{p2} = 40$, $k_{i1} = k_{i2} = 5$, and $k_{d1} = 10$ and $k_{d2} = 5$ for the ADP.

Figures 3 to 5 show the results of the simulations performed in JS and Simscape for the ASP, AIP, and ADP systems, respectively, under the two controlled scenarios described previously. Each figure shows the time evolution of the positions of the moving elements of each system, allowing a direct comparison between both results. In addition, internal plots illustrate the differences between the position signals obtained from the two simulations. In all cases, it is observed that the results obtained with the JS simulations present minimal differences with respect to those obtained by Simscape, especially during the execution of position regulation tasks. These discrepancies, for the most part, are not significant from the point of view of the overall dynamic behavior of the system. Based on these results, the simulations implemented in JS constitute an effective and sufficiently accurate alternative to represent the dynamic behavior of the electromechanical systems analyzed.

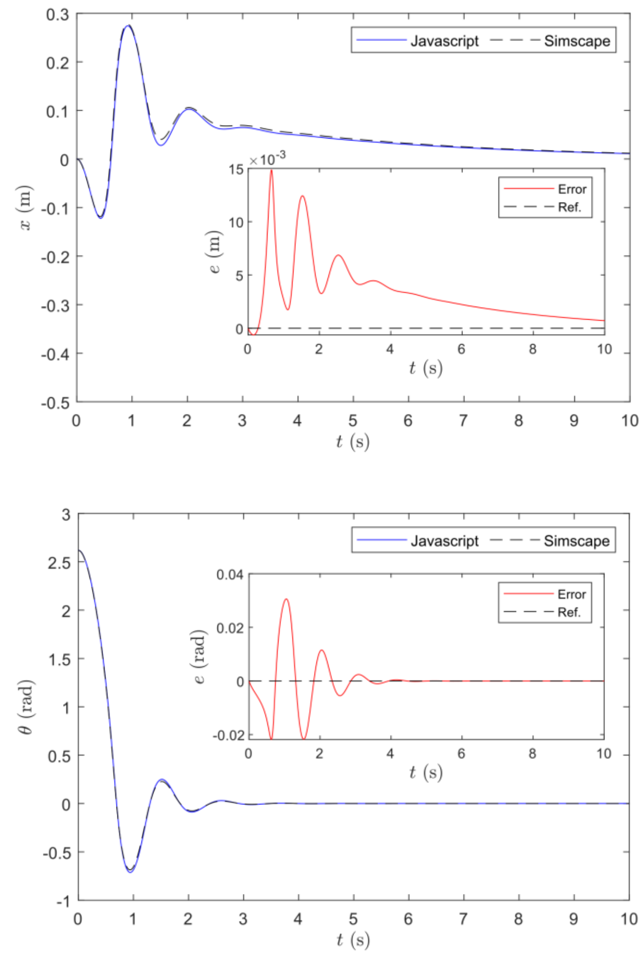


(a) Position of the pendulum under the first scenario

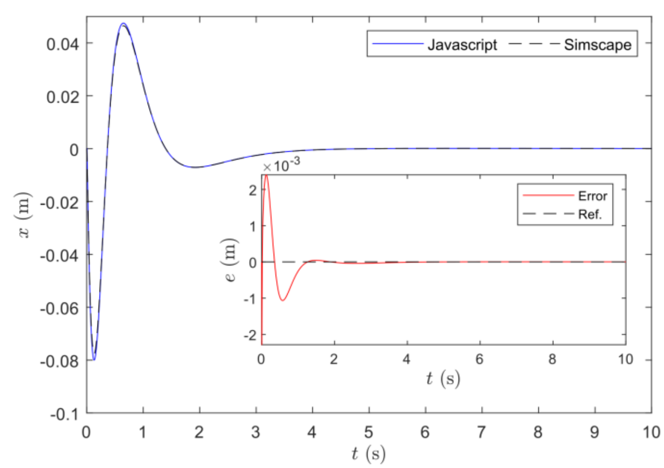


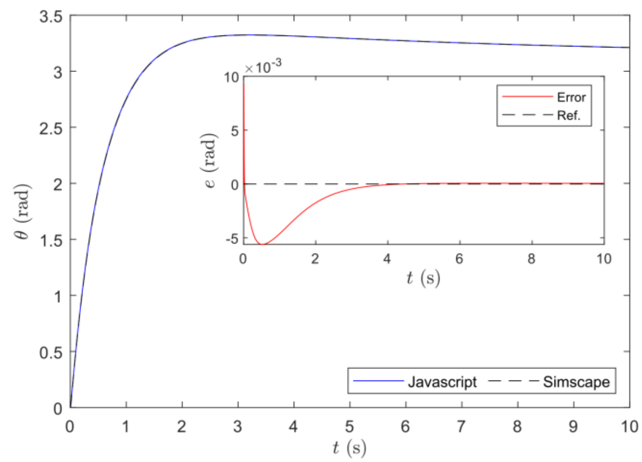
(b) Position of the pendulum under the second scenario

Fig. 3. Simulation results for the ASP under two controlled scenarios.



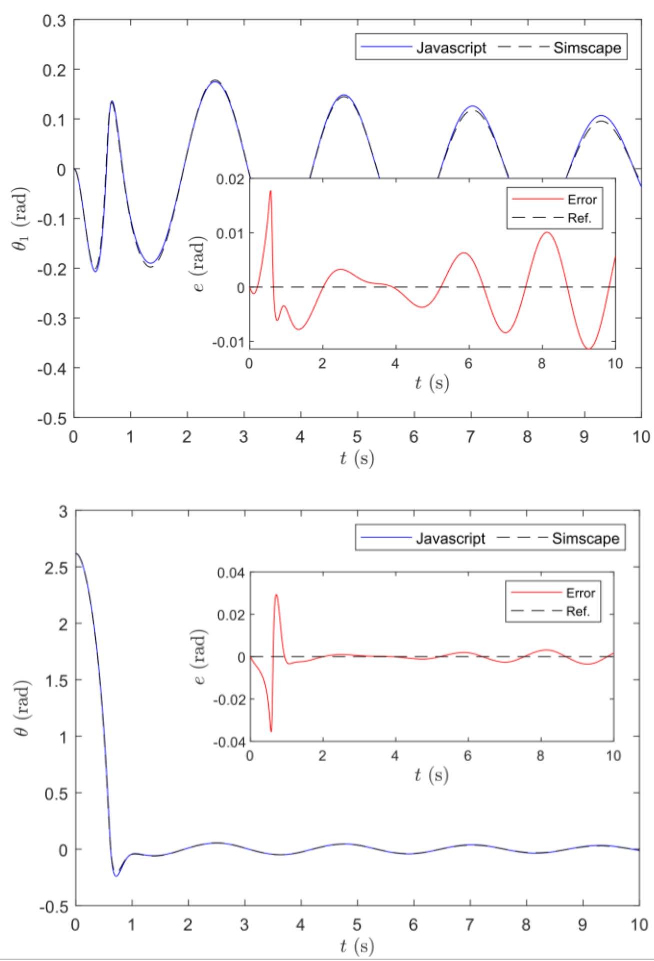
(a) Positions of the cart and pendulum under the first scenario



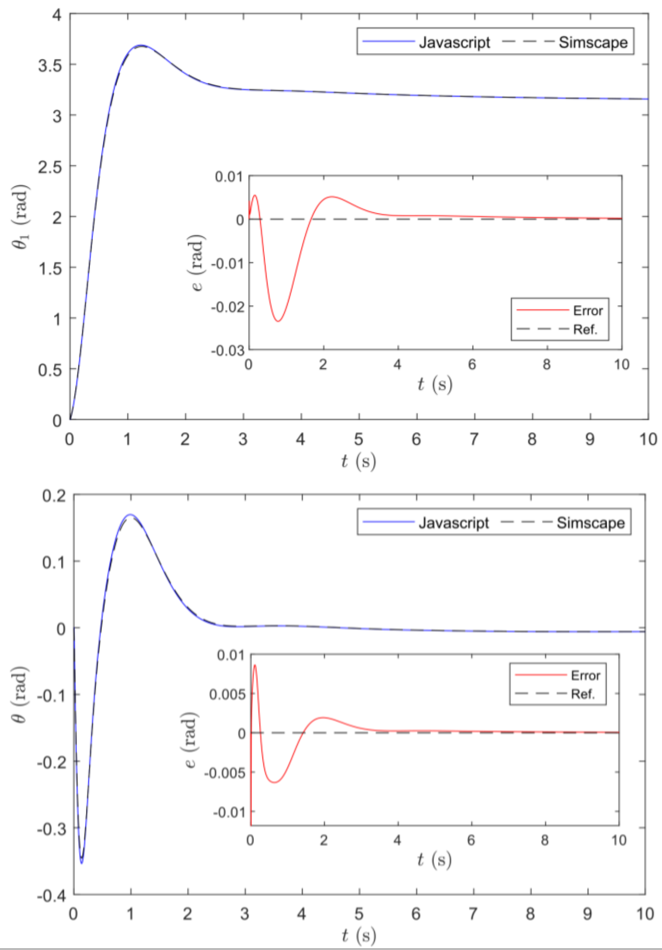


(b) Positions of the cart and pendulum under the second scenario

Fig. 4. Simulation results for the AIP under two controlled scenarios.



(a) Behavior of the positions of the two pendulums under the first scenario



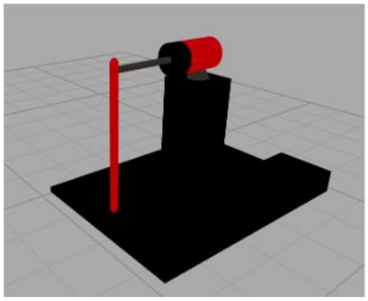
(b) Behavior of the positions of the two pendulums under the second scenario

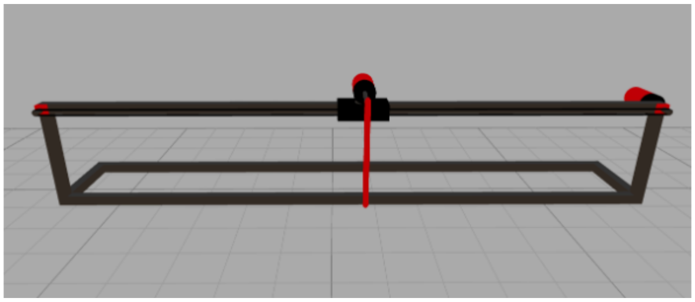
Fig. 5. Simulation results for the ADP under two controlled scenarios.

D. 3D modeling of the systems

The 3D models that visually represent the most relevant elements of each electromechanical system are shown in Fig. 3. These models were created using the free software tool Blender. In each of them, simple 3D geometries with basic materials showing only one color were used in order to reduce the computational load required to visualize them on the Web.

In the ASP model, shown in Fig. 6(a), the fixed base, electric motor, and pendulum are highlighted. For the AIP, illustrated in Fig. 6(b), the two electric motors, the rigid frame of the system, the pendulum, the moving base and the band that induces its motion are included. The ADP in Fig. 6(c) includes the base and the two pendulums with their respective electric motors. During the modeling of these systems, the moving elements were identified with representative names. In the case of the ASP, the only moving element is the pendulum. In the AIP, it is possible to move the base together with all the elements located on it, i.e. the pendulum and the electric motor that controls it, while in the ADP, both pendulums are moving, the second along with its electric motor.


(a) Simple pendulum


(b) Inverted pendulum

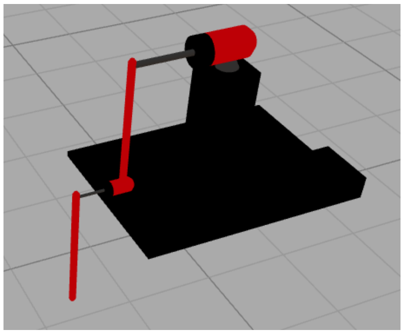

(c) Double pendulum

Fig. 6. 3D models of electromechanical systems.

E. 3D modeling of the platform

The 3D models of the ASP, AIP, and ADP systems, developed in the previous stage, are integrated into a single virtual space using Blender. In this space, other virtual objects that are familiar to users of the real platform are added. By distributing all objects, it is ensured that their dimensions, aspects and locations are consistent with reality. This arrangement of objects is shown in Fig. 7 and originates from a Web-optimized glTF file exported from Blender.

Fig. 7. 3D model of the platform.

F. Web site development

In this stage, the Web site shown in Fig. 8 is implemented, which will host the virtual platform of the electromechanical systems. Web technologies such as HTML, CSS and JS were used for its development, together with the Bootstrap and JQuery frameworks to speed up the process [38]. In the figure, several elements of the user interface can be seen, organized in a form that allows configuring different aspects of the simulation of each system, such as the initial conditions, the desired positions for the regulation task, and the control gains. In addition, buttons are included to start and stop the simulation of the systems. The user interface also features a simple navigation bar that includes a title and a button to show or hide the configuration form. Importantly, a space has been reserved in the interface for rendering the virtual platform.

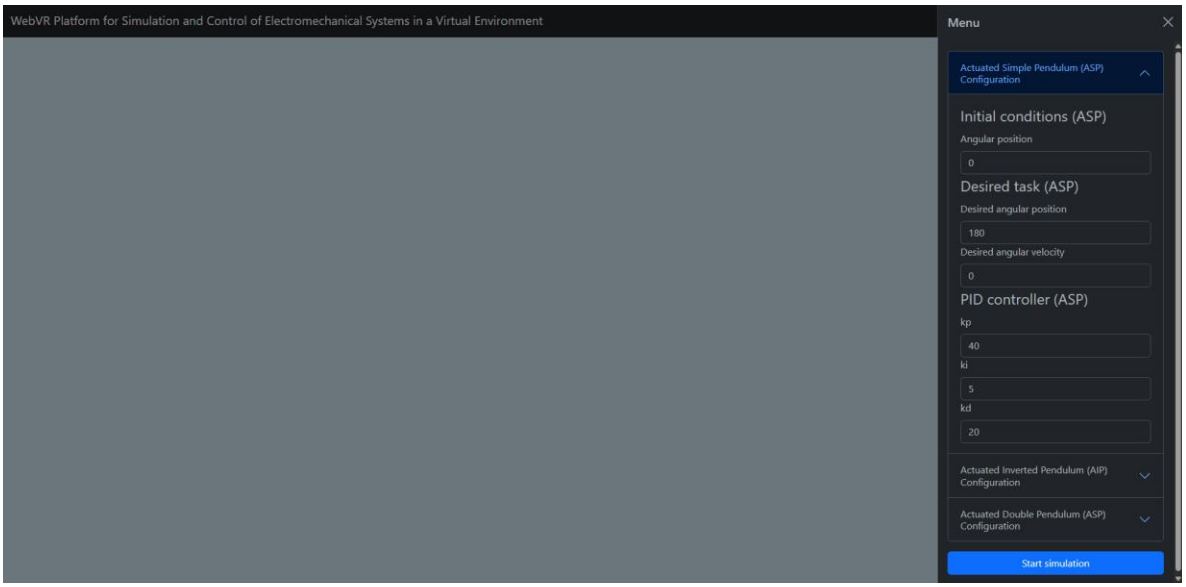


Fig. 8. Personalized Web site.

G. Virtual simulation integration

In the developed Web site, the dynamic simulation code and the virtual representation of the platform are integrated and coordinated.

To handle all aspects related to the virtual space, Three.js [39], a JS library that simplifies the operations required to display and manipulate 3D graphics on the Web, is used. At this stage, Three.js is used to import the glTF file from the platform, maneuver the elements within the virtual space (such as the moving components of the virtual electromechanical systems), and manage other elements essential for the successful rendering of the virtual environment on the Web site. Among these components are the lights, which illuminate the scene to highlight the colors and textures of objects and create 3D depth perception. Other key elements are the camera and renderer. The camera acts as the viewpoint of the virtual environment, determining how the scene is projected onto a 2D surface. The renderer, in turn, uses the information from the camera to draw the 2D projection of the scene as an image within the Web site. The Three.js editor (accessed April 09, 2025) was used to preview various configurations of these elements. During the loading of the virtual platform, JS references are created for the moving elements of the ASP, AIP, and ADP, which will later be manipulated using Three.js.

The JS code that encapsulates the functionalities of the dynamic simulation of electromechanical systems is imported directly into the site. Next, scheduled tasks are created in JS for each of the systems. These scheduled tasks repeatedly call a function or code block at fixed time intervals. Within each task, first a step forward in the corresponding system dynamics, i.e., the next state of the system, is calculated. Immediately afterwards, the state information is used to adjust the positions and orientations of the moving parts in their 3D representation with the help of Three.js. In order for the changes in the virtual space to be reflected in real time, the waiting interval is set to dt, i.e., $0.005\ (s)$ for each system.

H. Implementation of interactive mechanisms

At this stage, functionality is added to the Web site to manage all user actions and interactions. The user is allowed to navigate through the virtual space and observe the behavior of the electromechanical systems from any angle. To achieve this, the camera viewpoint is adjusted using the orbital controls provided by Three.js. These controls allow orbiting (rotating around a point), zooming (zooming in and out), and panning (moving vertically or horizontally) operations.

Another functionality of the Web site allows the user to start or stop the simulation by means of two buttons. For both buttons, event handlers are implemented in JS that execute a specific task when the user presses them. When the start button is pressed, first the start button (which is originally visible) is hidden and the end button (which is initially hidden) is shown. Then, the values provided by the user through the input elements in the interface are acquired and preprocessed, i.e. transformed into numerical values, to set up the simulation that is started next. When the end button is pressed, it is hidden while the start button becomes visible again, and the update of the simulations is stopped.

I. Deployment and utilization

The Web site has been deployed on an inexpensive hosting service, and the result can be seen in Fig. 9. The platform is available to any user from an up-to-date Web browser on any computing device, at any time and from any location in Virtual platform (accessed April 09, 2025). The availability of the platform is limited only by the availability of the hosting service, which is generally active about 99% of the time.

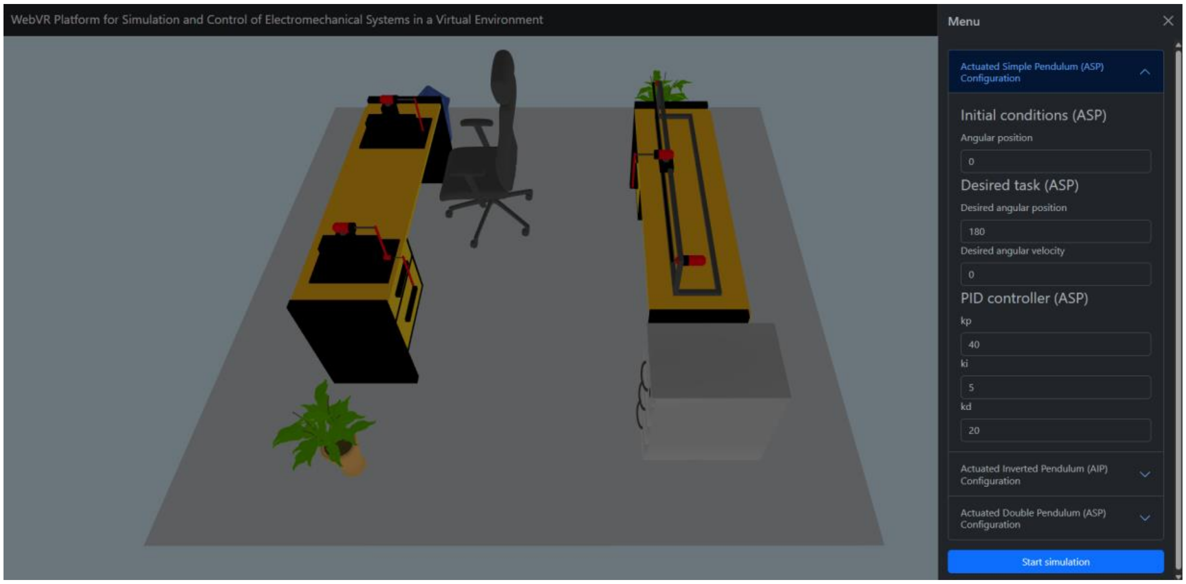


Fig. 9. Final platform deployed on a Web server.

IV. CONCLUSIONS AND FUTURE WORK

The proposed methodology for developing virtual platforms in WebVR has proven to be effective for the simulation of electromechanical systems, such as the simple pendulum, the linear inverted pendulum, and the double pendulum, all fully actuated. The resulting virtual platform allows performing numerical simulations of these systems, considering their kinematic, dynamic and control aspects, and visualizing the results in a 3D environment that reproduces elements similar to those of the real world. This type of platform is low-cost, does not require specialized virtual reality equipment, nor the installation of additional software beyond an updated Web browser. The methodology described can be applied to the development of platforms for other types of systems.

As future work, it is planned to include a wider variety of electromechanical systems in the virtual platform. In addition, the possibility of extending the platform developed in WebVR to an extended reality environment in WebXR will be explored. Finally, we will seek to incorporate mechanisms that allow collaborative interactions within the platform.

CRediT (*Contributor Roles Taxonomy*)

Author Contributions: M.A. **C.-M**.: Conceptualization, Data curation, Software, Visualization, Writing -- original draft, Writing - review & editing. A. **R.-M.**: Conceptualization, Methodology, Project administration, Supervision, Writing -- original draft, Writing - review & editing. G. **H.-H.**: Investigation, Methodology, Software, Visualization, Writing -- original draft, Writing - review & editing. M. **A.-P.**: Conceptualization, Funding acquisition, Project administration, Supervision, Writing -- original draft, Writing - review & editing. M.G. **V.-C.**: Investigation, Formal analysis, Funding acquisition, Validation, Writing -- original draft, Writing - review & editing. A.B. **R.-G.**: Data curation, Investigation, Software, Visualization, Writing -- original draft, Writing - review & editing.

Acknowledgments: Authors acknowledge the support from the Secretaría de Ciencia, Humanidades, Tecnología e Innovación (SECIHTI) of México.

Conflicts of Interest: The authors declare no conflicts of interest.

Data Availability Statement: Data available upon request.

Funding: No external funding was received for this work.

REFERENCES

[1] M. W. Ashraf, S. Tayyaba and N. Afzulpurkar, "Micro electromechanical systems (MEMS) based microfluidic devices for biomedical applications," *International journal of molecular sciences,* vol. 12, p. 3648–3704, 2011.

[2] A. Murakami and S. E. Lyshevski, Electromechanical systems, electric machines, and applied mechatronics, CRC press, 2018.

[3] Z. Li, L. Zheng, W. Gao and Z. Zhan, "Electromechanical coupling mechanism and control strategy for in-wheel-motor-driven electric vehicles," *IEEE Transactions on Industrial Electronics,* vol. 66, p. 4524–4533, 2018.

[4] E. Al-Hassan, H. Shareef, M. M. Islam, A. Wahyudie and A. A. Abdrabou, "Improved smart power socket for monitoring and controlling electrical home appliances," *IEEE Access,* vol. 6, p. 49292–49305, 2018.

[5] R. L. Williams, D. A. Lawrence and others, Linear state-space control systems, John Wiley & Sons, 2007.

[6] K. Ogata, Modern Control Engineering, 4th ed., USA: Prentice Hall PTR, 2001.

[7] R. P. Borase, D. K. Maghade, S. Y. Sondkar and S. N. Pawar, "A review of PID control, tuning methods and applications," *International Journal of Dynamics and Control,* vol. 9, p. 818–827, 2021.

[8] M. G. Villarreal-Cervantes, A. Rodríguez-Molina, C.-V. Garcia-Mendoza, O. Penaloza-Mejia and G. Sepúlveda-Cervantes, "Multi-objective on-line optimization approach for the DC motor controller tuning using differential evolution," *IEEE Access,* vol. 5, p. 20393–20407, 2017.

[9] A. Rodríguez-Molina, E. Mezura-Montes, M. G. Villarreal-Cervantes and M. Aldape-Pérez, "Multi-objective meta-heuristic optimization in intelligent control: A survey on the controller tuning problem," *Applied Soft Computing,* vol. 93, p. 106342, 2020.

[10] P. Brey, "Virtual reality and computer simulation," *The handbook of information and computer ethics,* p. 361–384, 2008.

[11] J. Xiong, E.-L. Hsiang, Z. He, T. Zhan and S.-T. Wu, "Augmented reality and virtual reality displays: emerging technologies and future perspectives," *Light: Science & Applications,* vol. 10, p. 1–30, 2021.

[12] N. Kapilan, P. Vidhya and X.-Z. Gao, "Virtual laboratory: A boon to the mechanical engineering education during covid-19 pandemic," *Higher Education for the Future,* vol. 8, p. 31–46, 2021.

[13] K. H. Cheong and J. M. Koh, "Integrated virtual laboratory in engineering mathematics education: Fourier theory," *IEEE Access,* vol. 6, p. 58231–58243, 2018.

[14] J. Gamo, "Assessing a virtual laboratory in optics as a complement to on-site teaching," *IEEE Transactions on Education,* vol. 62, p. 119–126, 2018.

[15] J. J. Brito, P. Toledo and S. Alayón, "Virtual laboratory for automation combining inventor 3D models and Simulink control models: Virtual laboratory for automation," in *2018 IEEE Global Engineering Education Conference (EDUCON),* 2018.

[16] B. M. Mîndruț and C. A. Oprea, "Experiential Learning Through Controlling and Monitoring a Real-Time 3D House Using LabVIEW in a Virtual Laboratory," in *International Conference on Innovative Technologies and Learning*, 2020.

[17] I. D. Wahyono, H. Putranto, K. Asfani, A. N. Afandi and others, "VLC-UM: A Novel Virtual Laboratory using Machine Learning and Artificial Intelligence," in *2019 International Seminar on Application for Technology of Information and Communication (iSemantic),* 2019.

[18] J. Ramírez, D. Soto, S. López, J. Akroyd, D. Nurkowski, M. L. Botero, N. Bianco, G. Brownbridge, M. Kraft and A. Molina, "A virtual laboratory to support chemical reaction engineering courses using real-life problems and industrial software," *Education for Chemical Engineers,* vol. 33, p. 36–44, 2020.

[19] D. A. Bastidas, L. F. Recalde, P. N. Constante, V. H. Andaluz, D. E. Gallegos and J. Varela-Aldás, "Non Immersive Virtual Laboratory Applied to Robotics Arms," in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, 2022.

[20] K. W. E. Cheng and C. L. Chan, "Remote hardware controlled experiment virtual laboratory for undergraduate teaching in power electronics," *Education Sciences,* vol. 9, p. 222, 2019.

[21] B. Hasan, Y. Al-Quorashy, S. Al-Mousa, Y. Al-Sahhaf and M. El-Abd, "V-LAB–the virtual electric machines laboratory," in *2020 IEEE Global Engineering Education Conference (EDUCON),* 2020.

[22] S. Su, Z. Wang, S. Fu and D. Huang, "Compensation Method of Flight Simulator Visual System," in *International Conference on Human-Computer Interaction,* 2022.

[23] M. Dombrowski, R. Buyssens and P. A. Smith, "Virtual reality training to enhance motor skills," in *International Conference on Virtual, Augmented and Mixed Reality*, 2018.

[24] R. Pasteka, J. P. Santos da Costa, N. Barros, R. Kolar and M. Forjan, "Patient–Ventilator Interaction Testing Using the Electromechanical Lung Simulator xPULM™ during V/AC and PSV Ventilation Mode," *Applied Sciences,* vol. 11, p. 3745, 2021.

[25] S. C. Ferreira, R. O. Chaves, M. C. d. R. Seruffo, A. Pereira, A. P. D. S. Azar, Â. V. Dias, A. d. A. S. d. Santos and M. V. H. Brito, "Empirical evaluation of a 3D virtual simulator of hysteroscopy using Leap Motion for gestural interfacing," *Journal of Medical Systems,* vol. 44, p. 1–10, 2020.

[26] E. B. S. Lustosa, D. V. de Macedo and M. A. F. Rodrigues, "Virtual simulator for forklift training," in *2018 20th Symposium on Virtual and Augmented Reality (SVR)*, 2018.

[27] O. A. Aguirre, J. C. Ñacato and V. H. Andaluz, "Virtual simulator for collaborative tasks of aerial manipulator robots," in *2020 15th Iberian Conference on Information Systems and Technologies (CISTI)*, 2020.

[28] R. Badarudin and D. Hariyanto, "Visual validation of PLC program using virtual simulator," in *Journal of Physics: Conference Series*, 2021.

[29] C. Li, L. Fu and L. Wang, "Innovate engineering education by using virtual laboratory platform based industrial robot," in *2018 Chinese Control And Decision Conference (CCDC)*, 2018.

[30] A. A. Sutchenkov and A. I. Tikhonov, "Electrical engineering materials virtual laboratory," in *2018 IV International Conference on Information Technologies in Engineering Education (Inforino)*, 2018.

[31] J. M. Ramírez-Romero and S. Rivera-Rodríguez, "Characteristics and functions of a virtual laboratory of induction machines in the teaching environment," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje,* vol. 13, p. 130–135, 2018.

[32] L. Ljung, "System Identification," in *Signal Analysis and Prediction*, A. Procházka, J. Uhlíř, P. W. J. Rayner and N. G. Kingsbury, Eds., Boston, MA: Birkhäuser Boston, 1998, p. 163–173.

[33] T. Mullen, Mastering blender, John Wiley & Sons, 2011.

[34] A. Ranjan, A. Sinha and R. Battewad, JavaScript for modern web development: building a web application using HTML, CSS, and JavaScript, BPB Publications, 2020.

[35] M. A. Johnson and M. H. Moradi, PID control, Springer, 2005.

[36] K. Ogata, System dynamics / Katsuhiko Ogata., 4th ed. ed., Upper Saddle River, NJ: Pearson/Prentice Hall, 2004.

[37] A. Rodríguez-Molina, J. Solís-Romero, M. Á. Paredes-Rueda, C. A. Guerrero-León, M. E. Mora-Soto and A. H. Herrera, "Exploiting evolutionary computation techniques for service industries," in *Evolutionary Computation with Intelligent Systems*, CRC Press, 2022, p. 153–179.

[38] P. Hong, Practical web design: Learn the fundamentals of web design with HTML5, CSS3, bootstrap, jQuery, and vue. js, Packt Publishing Ltd, 2018.

[39] B. Danchilla and B. Danchilla, "Three. js framework," *Beginning WebGL for HTML5,* p. 173–203, 2012.