Acceleron Aerospace Journal || AAJ.11.2106-2507
Volume 4, Issue 1, pp (872-885)
Research Paper
E-ISSN- 2583-9942 || DOI: 10.61359/11.2106-2507

Integrating Python and MATLAB for Digital Twin Applications in Aeroservoelasticity

K. Sudha Deepthi[1*], Kartik S Tandel[1†], Rammohan B[2‡], Kamal Babu P[3§]

[1]*Department of Mechanical Engineering, Dayananda Sagar University, Bengaluru, Karnataka 562112, India*

[2]*Department of Aerospace Engineering, Dayananda Sagar University, Bengaluru, Karnataka 562112, India*

[3]*School of Computer Science and Engineering, RV University, Mysuru Road, Bengaluru, Karnataka 560059, India*

[4]*Department of Mechanical Engineering, Dayananda Sagar University, Bengaluru, Karnataka 562112, India*

Abstract: The development of Digital Twin (DT) applications in intricate engineering areas has been greatly accelerated by the incorporation of sophisticated computational tools like Python and MATLAB. The goal of this research is to model, simulate, and analyze aeroservoelastic systems for Digital Twin implementations by utilizing the combined powers of Python and MATLAB. In order to achieve the real-time predictive skills needed for Digital Twins, aeroservoelasticity—which involves the coupled interplay of aerodynamic, structural, and control forces—presents special obstacles. This work successfully tackles these issues by fusing MATLAB's strong numerical and control system capabilities with Python's adaptability and library ecosystem. Using real-time sensor data streaming, and feedback control system implementation is used in this work. High-fidelity state-space modeling, sophisticated control design (such as PID and state feedback controllers), and system dynamics visualization are all done concurrently with MATLAB. In order to simulate and stabilize a highly flexible aeroelastic system, a continuous feedback loop was modeled in both environments, highlighting the platforms' complimentary advantages. Through the MATLAB Engine API for Python, the integrated method facilitates smooth data transmission between Python and MATLAB, guaranteeing real-time communication and model parameter synchronization. The efficiency of this dual-platform system is demonstrated by case studies on wing flutter suppression, limit cycle oscillation (LCO) mitigation, and control input optimization. The method is well suited for the dynamic needs of aeroservoelastic Digital Twins because of the results, which demonstrate a shorter error convergence time, improved stability, and computational economy. By utilizing MATLAB and Python's interoperability, this work establishes the groundwork for scalable, real-time Digital Twin solutions in aeronautical engineering. It highlights how crucial hybrid computational ecosystems are for bridging the gap between high- fidelity simulations and real-time predictive capabilities, opening the door for improvements in fault prediction, control optimization, and aeronautical system monitoring.

Table of Contents

[*]Department of Mechanical Engineering, Dayananda Sagar University, Bengaluru, Karnataka 562112, India.
✉ **Corresponding Author: sudha-bosch@dsu.edu.in. [ORCID ID: https://orcid.org/0009-0006-5305-9876]**
[†]Department of Aerospace Engineering, Dayananda Sagar University, Bengaluru, Karnataka, 562112, India.
Contact: kartikmec@gmail.com. [ORCID ID: https://orcid.org/0000-0002-6645-5011]
[‡] School of Computer Science and Engineering, RV University, Mysuru Road, Bengaluru, Karnataka, 560059, India.
Contact: rammohanbhanumurthy@gmail.com. [ORCID ID: https://orcid.org/0000-0002-6254-5546]
[§] Department of Mechanical Engineering, Dayananda Sagar University, Bengaluru, Karnataka,562112, India.
Contact: kamalbabu-me@dsu.edu.in. [ORCID ID: https://orcid.org/0000-0002-1622-9542]
** Received: 25-January-2025 || Revised: 29-January-2025 || Accepted: 29-January-2025 || Published Online: 30-January-2025

1. Introduction

A A digital twin is a computerized model of a real asset, procedure, or system that facilitates data monitoring and analysis to produce better business results. In the production technology sector, the idea of a "digital twin" first surfaced as a way to build a computer model of a physical process that was linked to the actual object by a "digital thread." Through the use of this digital thread, the digital twin was able to continuously receive data from physical sensors that recorded the machine's condition. For a specific purpose, the digital twin combines data, models, and other information about the physical object created during the course of its existence. Having a digital representation that is appropriate for the task at hand in terms of accuracy, completeness, speed of execution, and amount of detail is the goal of the digital twin. The digital twin can be used for a variety of physical items, such as industrial processes, infrastructural systems, process plants, and goods. The idea of the digital twin has also spread outside of the manufacturing sector and is currently being used in smart cities and healthcare, among other fields. Better product design increased operational effectiveness, and more efficient maintenance techniques are some of the main advantages of digital twins **[1]**.

The term "digital twin" refers to a physical system, process or asset in general; Modeling and simulation are becoming common practice in system development, such as when developing and verifying task. All parts of the life cycle are included in the simulation. The current method assists operators and software/hardware developers in understanding the complexity of mechatronic systems. Software and network communication expand the capability of mechatronic systems as the technological hurdles rise. Their linkages will become increasingly entwined as the conventional mechatronic disciplines are recognized in a more integrated manner. Multi-domain and multi-level simulation techniques are required to design these systems, validate their attributes through early-stage virtual testing, and support their operation and service delivery. The process of system engineering and development must incorporate these methods. The digital twin requires its own architecture and should have its basic structure planned out in advance. It expands the pure data that is accessible in design and engineering and that is gathered by simulation models during operation and service. These simulation models include system descriptions, performance ratings, and quality concerns **[2][3]**. The digital twin maintains consistency by acting as an interface for various models and data at various granularities. enhancing mechatronic systems and products while they are being used or operated. It is necessary to close the gap between development and operation. The redesign of operational and service paradigms is time- and money-consuming. The digital twin links various value chains. For example, in a production system, the digital twin of a product is used to transport crucial information to producers and manufacturers. This requires a digital twin that contains models with various levels of granularity and, in general, calls for the predefinition of its primary architecture. A digital twin is a virtual representation of a system integrating the appropriate physical models, sensor data, fleet history etc. It is an integrated, multiphysics multiscale model of a physical asset. The digital twin is accurate and may take into account one or more systems or sub-systems of a physical asset. Additionally, the digital twin continually monitors the health of the system can perform and prognosis including computation of its remaining usable life **[4]**. A component, product, or system's complete physical and functional description that includes essentially all the information that may be helpful at different stages of its lifespan is referred to as a "Digital Twin" in the generic sense. Technically speaking, this is not possible. The amount of data is too large, too varied, and entirely unstructured. Additionally, new applications in later phases need particular data and information preparation in earlier phases. It is required for a Digital Twin to have a certain architecture. The term "Digital Twin" refers to a description of a part, a product, or a system provided by a collection of cohesive executable models that have the following properties.

- The Digital Twin is a connected set of pertinent digital artefacts that includes engineering data, operational data, and descriptions of behavior through various simulation models. The simulation models that make up the Digital Twin are tailored for the task at hand and employ the appropriate level of accuracy.
- The Digital Twin incorporates the most recent information about the real system and grows with it during its entire life cycle.
- Digital Twin provides functions for assist systems to help them maximize operation and service, in addition to being used to explain behavior and create solutions applicable to the real system. The idea of Model-Based Systems Engineering (MBSE) is therefore expanded by the digital twin from the engineering and manufacturing stages to the operation and service phases **[5]**.

The Digital Twin serves as the foundation for simulation-driven assist systems, control, and servicing decisions made while the system is in operation. There are three different kinds of systems: basic (simple), complex, and complicated. Simple systems are those in which the inputs are visible and the results are easily

examined. Complex systems are predictable and contain more components whose inputs are well understood. Complicated systems feature a wide network of parts, multiple communication routes, and a vast amount of information, making it difficult to predict their behavior. The system's behavior is divided into two categories: Predicted Behavior and Unpredicted Behavior. These two categories are further divided into four groups: Predicted Desirable Behavior, Predicted Undesirable Behavior, Unpredicted Desirable Behavior, and Unpredicted Undesirable Behavior. The primary requirement of the system, which is designed to fulfill specific criteria, is the Predicted Desirable Behavior. Unpredicted Desirable Behavior refers to scenarios where the system's actions cannot be fully anticipated, whereas Unpredicted Undesirable Behavior highlights critical issues that require additional solutions **[6], [7]**.

For the benefits of the Digital Twin to be realized on a larger scale, numerous complex issues are created and identified by it. The Digital Twin has the power to transform the traditional approach to digital modeling and simulation. The industrial automation and manufacturing sectors as a whole could be completely redefined by the Digital Twin. The primary objective of a Digital Twin is to integrate the virtual and physical worlds of an engineering model so that it can test, simulate, and assess the model's design. The entire engineering process is represented by the Digital Twin, which can effectively interface the physical and virtual realms. The biggest challenge with Digital Twins is the interfacing of data and multi-physics models, which necessitates a highly efficient framework for drive and control. A Digital Twin is also used to optimize the product lifecycle by bridging the gap between the virtual and physical worlds and by analyzing big data and real product lifecycle metrics. Various actions in the product lifecycle can be controlled, simulated, optimized, and assessed in the virtual realm of the Digital Twin in the same manner. The new engineering method incorporates "Virtual Twins," which function offline during the design phase but have digital counterparts based on data gathered during online operations. Real-time solutions for physically based models present a key challenge that must be addressed by integrating virtual and physical domains within a design model **[8]**.

2. Integrating Python and MATLAB for Digital Twin Applications in Aeroservoelasticity

The challenges of modeling, simulating, and analyzing aeroservoelastic systems are addressed by combining the advantages of two powerful programming environments through the integration of Python and MATLAB in digital twin applications for aeroservoelasticity. The relationship between aerodynamic forces, structural dynamics, and control systems is known as aeroservoelasticity, and it is especially important in high-performance aerospace systems such as wind turbines and airplanes. Digital twins are essential for improving system performance and reliability because they are virtual copies of physical systems that provide real-time analysis and predictive insights **[9]**. In the ever-changing field of aeroservoelasticity, the development of digital twin applications has become increasingly crucial in addressing the complex challenges posed by the dynamic interaction of structural, control, and aerodynamic forces. To tackle these challenges, researchers have determined that a synergistic approach, leveraging the benefits of multiple programming languages and computational tools, is required. One such integration that shows promise is the use of Python and MATLAB—two powerful platforms with complementary features.

Python provides an ideal environment for developing algorithms and implementing advanced numerical techniques due to its robust software engineering capabilities and growing recognition in the scientific computing community. On the other hand, MATLAB's long-standing reputation in the scientific and technical fields, combined with its user-friendly interface and extensive collection of specialized toolboxes, makes it an attractive choice for rapid application development and prototyping. Current research explores the benefits of integrating Python and MATLAB, demonstrating how Python can extend MATLAB's capabilities, enabling the development of increasingly complex digital twin models and simulations. The integrated framework supports both data-driven insights and physics-based modeling, combining MATLAB’s computational robustness with Python’s versatility. Python is open-source and highly adaptable, offering a vast array of libraries for data analysis and machine learning (such as NumPy, SciPy, and TensorFlow). It also provides smooth interaction with cloud platforms for big data and Internet of Things (IoT) applications, making it an excellent choice for advanced visualization and data-driven methodologies. Meanwhile, MATLAB offers powerful tools for control system design and numerical computing, featuring reputable libraries for aerospace applications, such as the Aerospace Toolbox and Simulink. Additionally, MATLAB excels at handling simulations, matrix operations, and embedded system toolchain integration **[10]**.

2.1. Integration of Industry 4.0 with MATLAB

For the development of intelligent, automated, and data-driven systems, MATLAB's integration with Industry 4.0 (Figure 1, Source: Bosch Rexroth) concepts is highly relevant. MATLAB offers powerful modeling, simulation, and analytical capabilities that align well with Industry 4.0's emphasis on automation, big data analytics, cyber-physical systems, IoT, and AI/ML. It facilitates IoT integration and enables real-time data collection and visualization. The MATLAB IoT Toolbox can connect IoT sensors and devices, allowing for real-time data acquisition, processing, analysis, and visualization. In this study, Industry 4.0 focuses on the use of digital twins to replicate and monitor physical assets in real time. MATLAB and Simulink enable the creation of digital twin models for physical system modeling based on physics by incorporating real-world sensor data. These digital twins aid in process optimization, fault diagnostics, and predictive maintenance. By integrating MATLAB with Industry 4.0, it becomes possible to develop sophisticated, intelligent systems due to its robust toolchain, which spans the entire lifecycle—from data collection to simulation, optimization, and deployment **[11]**.

Figure-1 Industry 4.0 Kit [Image Source: Bosch Rexroth]

The system can be enhanced with Industry 4.0 by utilizing real-time monitoring and IoT. Sensors that measure displacement, velocity, and aerodynamic forces in real time, such as strain gauges and accelerometers, play a crucial role. MATLAB is used to generate control signals and analyze data. The behavior of the wing is replicated in MATLAB through a digital twin of the aeroelastic system, which incorporates real-world sensor data to model and predict future responses. By continuously monitoring the wing's reaction, irregularities indicating potential issues can be detected at an early stage. The primary advantage of this integration is its dynamic adaptation to structural modifications and aerodynamic forces, seamlessly combining with Industry 4.0 tools for predictive maintenance and real-time monitoring. The controller was designed using MATLAB's PID tuner to simulate the aeroelastic system both with and without the PID controller. The interaction between aerodynamic forces and structural flexibility makes aeroelastic systems—such as airplane wings—susceptible to instability. This instability can lead to Limit Cycle Oscillations (LCOs), flutter, and other issues. By integrating MATLAB with Industry 4.0 concepts—such as predictive analytics, real-time monitoring, and IoT—we can develop a feedback control system that minimizes displacement and velocity errors, maintains aerodynamic stability, and dynamically suppresses oscillations **[12]**.

2.2. Framework for Integration

The following elements are involved in the integration of MATLAB and Python in the creation of digital twins for aeroservoelasticity:

2.2.1. Data Acquisition and Preprocessing

Python is used for feature extraction, data cleaning, normalization, and data collecting from sensors, actuators, and other Internet of Thing's devices. and putting machine learning algorithms into practice for preliminary insights based on data.

2.2.2. Physics-Based Modeling and Simulation

MATLAB is used to simulate control methods and aeroelastic phenomena using Simulink, and to create physics-based models of aeroservoelastic systems using computational fluid dynamics (CFD) and finite element analysis (FEA).

2.2.3. Integration

To transfer data between Python and MATLAB, use file-based interfaces (such as.mat files or.csv files) or libraries like the MATLAB Engine API for Python. MATLAB can leverage APIs to call Python for more intricate analytics, or Python can initiate MATLAB simulations. For 3D charting and subsystem visualization, the visualization tools are MATLAB and Matplotlib for Python **[13]**.

2.3. Applications in Aeroservoelasticity

- ***Real-Time Monitoring:*** Sensors on aeronautical structures provide real-time data to Python, while MATLAB analyzes the data by comparing it with simulation results to detect anomalies.
- ***Predictive Maintenance:*** Machine learning techniques, such as support vector machines and neural networks, are implemented in Python to predict wear and failure scenarios. MATLAB analyzes vibration and stress data to enhance the accuracy of these predictions.
- ***Control System Optimization:*** MATLAB models control systems to mitigate flutter and Limit Cycle Oscillations (LCO). Python utilizes evolutionary techniques and machine learning to optimize control parameters.
- ***System Validation and Updating:*** Python processes real-time sensor data to update the data-driven models of the digital twin. MATLAB then incorporates these updated inputs to refine physics-based models **[14]**.

To elaborate it more detailed considering a digital twin for a wing structure, for a high-aspect- ratio wing in a wind tunnel undergoing LCO. The role of Python is to gather strain gauge and accelerometer data in real time. To forecast critical velocity for LCO, the machine learning models will be trained. The patterns and irregularities are visualized in an interactive manner. The MATLAB's Function is the use of the Aeroelastic Toolbox for conducting the frequency and flutter analysis, verifying outcomes using Simulink numerical simulations and improving the system's parameters to increase the accuracy of predictions **[15]**.

The Advantages of Python-MATLAB Integration are the Cost-effectiveness: Python's open- source status lowers the initial software development costs. Improved analytical depth: Python's machine learning insights enhance MATLAB's flexible and rigorous simulations. Time savings and frameworks that are simple to modify for various aerospace applications parallel execution of programs such as MATLAB simulation and Python data processing. Data compatibility, which guarantees consistent data formats and library usage, performance overhead, which maximizes inter-software communication and reduces redundant computations, and the Learning Curve, which also trains teams to manage dual- environment workflows, are the problems and their solutions. For digital twin applications in aeroservoelasticity, combining Python and MATLAB is a potent and complementary strategy. It combines MATLAB's accuracy in numerical and control system analysis with Python's adaptability in data analytics and machine learning. This collaboration enhances the potential of digital twins in aircraft engineering by enabling real-time monitoring, predictive maintenance, and optimization of aeroservoelastic systems **[16]**.

2.4. Hybrid Modeling Approaches

The requirement for precise and effective modeling of the underlying physical systems is a crucial component of the creation of digital twins. This entails integrating high-fidelity aerodynamic, structural, and control models in the context of aeroservoelasticity. By combining data-driven methodologies with physics-based models, hybrid modeling approaches have become a viable way to overcome the drawbacks of conventional modeling techniques. These hybrid techniques enhance online learning, generalizability, and interpretability by combining the best features of data-driven and physics-based models. For the development of hybrid models for digital twin applications in aeroservoelasticity, MATLAB and Python offer some advantages. While physics-based models can be combined using MATLAB's extensive library of specialized toolboxes and numerical solvers, Python's powerful data analysis and machine learning tools can be used to create data-driven components **[17], [18]**.

2.5. Verification and Validation

A vital aspect of the creation of digital twin models is guaranteeing their accuracy and dependability. Accurate forecasts and dependable simulations require strong verification and validation methods. Python and MATLAB together can provide a comprehensive framework for model validation and program verification. Python's unit testing and continuous integration capabilities can be used to ensure the codebase's accuracy and stability, while MATLAB's advanced visualization and simulation tools can be used to compare the performance of the digital twin models to real data **[19]**.

3. Methodology

By combining structural dynamics, aerodynamics, and control systems, the multidisciplinary area of aeroservoelasticity analyzes and maximizes the performance of flexible aerospace structures in dynamic situations. This field examines the interaction between aerodynamic forces, structural flexibility, and feedback control to ensure the stability and operation of aircraft, spacecraft, and other aerodynamic systems. Aeroservoelasticity investigates the relationship between structural elasticity and aerodynamic forces. One of the primary phenomena is flutter, a dynamic instability in which long-lasting oscillations are caused by aerodynamic forces. Divergence is another key phenomenon, and buffeting refers to random vibrations brought on by unstable aerodynamic loads. Static instability occurs when aerodynamic forces surpass structural restoring forces, resulting in catastrophic deformation. The field also investigates the effects of control surfaces, such as rudders, elevators, and ailerons, on aerodynamic forces and the structural reaction in relation to the dynamics of elastic structures **[20]**. It focuses on feedback loops between structural dynamics and control systems, combining servoelastic and aeroelastic effects, which is a crucial component in the design of active control systems to improve performance and reduce aeroelastic instabilities.

Flutter suppression is one application for aeroservoelasticity. By dampening flutter, active control systems (such as sensors and actuators) enable operation at higher speeds or dynamic pressures. Load alleviation enhances fatigue life and safety by lowering structural loads during gusty or turbulent conditions. Aeroservoelastic analysis helps flexible, lightweight unmanned aerial vehicles (UAVs) maintain stability and control under various situations, while adaptive control systems modify control surfaces to maximize lift, drag, and maneuverability. For aerospace structures, spacecraft with large, flexible appendages—like solar panels—require aeroservoelastic analysis for stability during launch and operation. Among the fundamental concepts of aeroservoelasticity is dynamic stability, where the system needs to maintain stability in the face of feedback control, aerodynamic, structural, and control interactions. Sensors, actuators, and controls are used to improve stability and actively control structural responses **[21]**. Modal analysis determines the structure's natural vibration modes to forecast its aeroelastic and servoelastic behavior, while gust response assesses the structure's reaction to abrupt variations in airflow and turbulence. To simulate the aeroservoelastic behavior, control system equations are combined with aeroelastic models. For structural dynamics, numerical methods such as Finite Element Analysis (FEA) are used. For aerodynamic forces, Computational Fluid Dynamics (CFD) is applied. State-space models integrate control systems, and solving the structural, aerodynamic, and control equations simultaneously predicts the system's response in various scenarios.

Experimental investigations in the field of aeroelasticity have served two major purposes. During flutter analysis, testing of wind tunnel models with properly scaled mass and stiffness properties has often been more rewarding than equivalent efforts using analytical techniques or even full-scale airplanes. Aeroelasticity is typically defined as the science of investigating the mutual interplay of aerodynamic and elastic forces, and how this relationship affects aircraft design. Aeroelastic problems would not exist if aircraft structures were entirely rigid. Modern airplane structures are extraordinarily flexible, which accounts for the wide range of aeroelastic phenomena. Structural flexibility may not be inherently undesirable, but aeroelastic events occur when structural deformations cause additional aerodynamic forces. These higher aerodynamic forces may generate further structural deformations, resulting in even greater aerodynamic forces. Such interactions may tend to shrink in size until a stable equilibrium is reached, or they may diverge and destroy the structure. Aeroelasticity is not a complete description because many key aeroelastic occurrences involve both inertial and aerodynamic/elastic forces. Aeroelasticity includes phenomena involving interactions among inertial, aerodynamic, and elastic forces, as well as other phenomena involving interactions between aerodynamic and elastic forces. The former will be referred to as dynamic, and the latter as static aeroelastic phenomena. The figure below depicts the interaction between three important fields of aerospace engineering: structural dynamics, unsteady aerodynamics, and automatic flight

control system dynamics. These areas interact to provide important disciplines such as aeroelasticity, servoelasticity, and aeroservoelasticity.

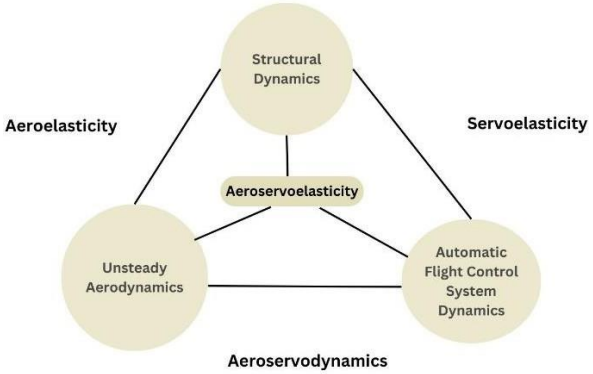

Figure-2 Block schematic of a typical aeroservoelastic closed-loop system [22]

As airplane speeds increased over time, structural limitations imposed by design criteria resulted in a stiff construction that could prevent most aeroelastic events. However, as aircraft designers faced the challenge of higher speeds, load demands didn't rise significantly, and no established stiffness guidelines were available for the design of flexible structures. This shift introduced a range of challenges in aerospace engineering. While aeroelastic phenomena have only gained their current prominence relatively recently, they have influenced airplane design since the onset of powered flight **[23]**. In modern high-speed aircraft, aeroelastic phenomena—particularly flutter—have significant effects on structural design. Flutter, one of the most impactful aeroelastic phenomena, occurs when aerodynamic forces induce oscillations in the structure, leading to instability. Classical flutter is associated with potential flow and typically involves the coupling of two or more degrees of freedom. Non-classical flutter, which is more complex, includes effects like divided flow, periodic flow reattachment, stalling conditions, and time-lag effects between aerodynamic forces and structural motion. Preventive measures often involve increasing stiffness or modifying mass distribution, or a combination of both. For instance, wing torsional stiffness is one of the most important stiffness characteristics affected by flutter. In this case, heavier mass components in the wing are often positioned to prevent flutter. Additionally, wing planform and aspect ratio significantly influence flutter characteristics. A decrease in aspect ratio and an increase in sweep generally lead to higher flutter speeds, whereas an increase in aspect ratio and a decrease in sweep, including forward sweep, reduce flutter speeds **[24], [25]**.

A practical formulation of aeroelastic problems is essential for understanding the flow conditions and structural characteristics. It is often useful to begin by considering the problem in a general form, introducing assumptions to specialize the analysis for particular cases. In aeroelastic analysis, elements such as wings or airframes can serve multiple functions. For instance, an airplane wing functions as a lift-producing surface, an elastic structure, and an inertia force element. The analysis of these functions can be approached through functional block diagrams, which visually represent the interactions among the system's various components **[26]**. The relation between $\boldsymbol{L}$ and $\boldsymbol{\alpha}$ depends on the elastic deformation of the wing. It is convenient to consider the wing as composed of two elements, the wing as a lift-producing mechanism, and the wing as an elastic structure. With respect to aerodynamics, each configuration of the wing may be considered as rigid. With respect to the elastic deformation, the action of aerodynamic forces is the same way as any other system of exterior forces. The partial problems in aerodynamics and elasticity may be solved in the classic manner. But these two solutions must be properly combined to account for the behaviour of an elastic wing. An elastic wing may be represented functionally in the figure below. The airfoil, at an angle of attack $\boldsymbol{\alpha}$ and having a deflection surface θ, the wing surface corresponding to the initial angle of attack $\boldsymbol{\alpha}$ also by the symbol α; then the geometrical configuration may be simply written as $\boldsymbol{\alpha} + \theta$, with respect to which the aerodynamic force is computed. Thus, the rigid-airfoil-elastic-structure system forms a loop which is a feedback system **[27]**. To conceptualize the feedback loop in the context of aeroelastic systems with state-space dynamics, the aircraft wing in a feedback control system the goal is to maintain stability and limit the flutter. Aerodynamic forces (lift and drag), structural flexibility (elastic forces), and actuator inputs (control surfaces like flaps or ailerons) interact to produce aeroelastic forces on the aircraft wing. From the equation below considering the aircraft wing as feedback control system. The aeroelastic system (e.g., an aircraft wing) can be modeled in MATLAB using state-space equations. The purpose of using MATLAB and Industry 4.0 concepts to stabilize an aeroelastic model of airplane wing is to preserve aerodynamic stability while dynamically reducing oscillations caused by structural flexibility. The goal is to suppress Limit Cycle Oscillations

(LCOs) in an aircraft wing caused by interactions between aerodynamic forces and structural flexibility. The mathematical representation of the dynamics of an aeroelastic system in terms of states such as displacement, velocity, and aerodynamic force is known as state-space modelling. Feedback Control System is the real-time stabilization of the system by the reduction of displacement and velocity errors through the use of a PID controller. Dynamic Suppression of LCOs is by simulating, optimizing, and implementing the control system for practical uses with MATLAB. State-space equations are used in MATLAB to simulate an aeroelastic system, such as an airplane wing **[28]**.

***Formulation* [29], [30]**

The state-space equations model the dynamics of the system that records the motion of the wing as a result of aerodynamic forces and structural flexibility.

$$\hat{x} = Ax + BQ_C \tag{1}$$

$$\dot{x}_C = A_C x_C + B_C u \tag{2}$$

Using the state vector for aeroelasticity:

$$x = \begin{pmatrix} q^T \\ q^T \\ x^T \\ x^T \end{pmatrix} \tag{3}$$

The actuators' state vector, X_g, and the gust state vector:

$$x_c = \begin{pmatrix} \delta^T \\ \delta^T \\ \xi^T \end{pmatrix} \tag{4}$$

The control forces:

$$Q_C = C_C x_C + D_C u \tag{5}$$

The output equation for the aeroelastic system, where the measured signals include wing displacement and velocity:

$$y = Cx + DQ_C \tag{6}$$

Linear feedback control rule:

$$u = \hat{k}\hat{x} + ky \tag{7}$$

The controller state equation:

$$\dot{\hat{x}} = F\hat{x} + Ly + Hu \tag{8}$$

The PID controller feedback rule:

$$u = k_0(\delta_c - \delta) + k_1(\dot{\delta}_c - \dot{\delta}) \tag{9}$$

The transfer function of the servo controller:

$$H(s) = k_0 + k_1 s \tag{10}$$

The servo actuator dynamics with combined actuator and PD control:

$$(I\delta - a_2)\ddot{\delta} + (k_1 - a_1)\dot{\delta} + (k\delta + k_0 - a_0)\delta = K_0\delta_c + k_1\dot{\delta}_c + (a_3, \ldots, a_n)\xi_c \tag{11}$$

The control aerodynamic state vector ξ_c is the solution to:

$$\dot{\xi}_c = \Lambda_c \xi_c + \Gamma_c \dot{\delta} \tag{12}$$

The actuator state vector is defined as:

$$x_c = \begin{pmatrix} \delta \\ \delta \\ \xi^T \end{pmatrix} \tag{13}$$

The closed-loop servo actuator state equation:

$$x_C = A_B x_C + B_C(k_0\delta_C + k_1\dot{\delta}_c) \tag{14}$$

The transfer function of this servo:

$$H(s) = \frac{k_0 + k_1 s}{s} \tag{15}$$

4. Results and Discussion

A key idea in control engineering is the feedback loop control system, in which a system's output is continuously observed, compared to a target setpoint (or reference), and modified to reduce the discrepancy (error). This method ensures that, despite disruptions or uncertainties, the system functions as planned. The reference or intended value for the system's output—such as the ideal position, temperature, or velocity—is known as the input. A proportional-integral-derivative (PID) controller is one that responds to step commands with zero steady-state error. The fundamental PD/PID control feedback technique is used to stabilize a servo actuator. Corrective aerodynamic forces are applied by ailerons or flaps in response to the control input. The proportional-integral-derivative (PID) controller is a popular feedback method. The control input is adjusted based on three criteria: Integral: removes errors in stable, long-term conditions; Derivative: forecasts future errors by accounting for the rate of change; Proportional: corrects errors instantly. In the context of this research project (aeroservoelastic systems), the feedback loop compensates for dynamic interactions by maintaining displacement, velocity, and control input constant around target levels. When the sensor measures the wing's present condition (velocity and displacement), the inaccuracy is detected. The difference between the intended and actual states is used to compute the error. In order to counteract disturbances and minimize inaccuracy, the actuators (such as flaps) apply forces. The loop continues until the mechanism stabilizes, while the sensor keeps track of the wing's updated condition. The benefits of the feedback system include flutter suppression, which dynamically dampens oscillations to prevent catastrophic flutter; robustness, which adjusts to changes in aerodynamic conditions (such as fluctuating wind speeds); and stability, which ensures that the wing operates within safe bounds and prevents excessive deformation. Sensors detect rising displacement and velocity fluctuations. To counteract the oscillations, the controller computes corrective forces. Aerodynamic forces are applied by actuators, such as ailerons, to reduce flutter. The wing is stabilized around the intended displacement by the mechanism. Using Python and MATLAB programming, a continuous feedback loop on the state-space dynamics and a PID controller are simulated for the aeroservoelastic system. The system effectively reduces displacement errors and suppresses velocity fluctuations by continuously modifying the control input in response to real-time feedback. To ensure system stability, the proportional, integral, and derivative components cooperate to remove long-term deviations, correct immediate errors, and predict future oscillations. By highlighting the crucial role that feedback loops play in reducing limit cycle oscillations and preserving structural integrity, this simulation offers insightful information on managing intricate aeroelastic systems. These methods are essential for aerospace applications where stability and precise control are critical.

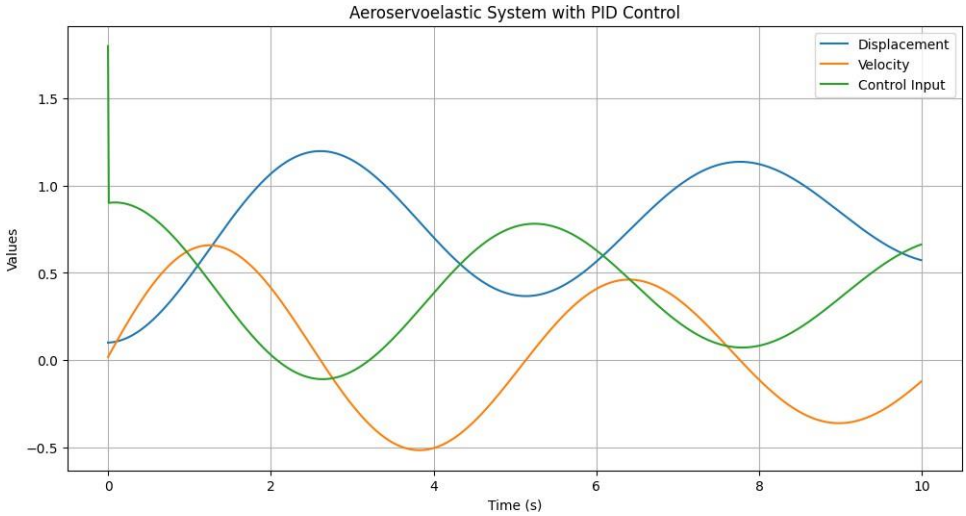

Figure-3 Dynamic Response of an Aeroservoelastic System with PID Control

The system's oscillatory motion, such as wing displacement, is represented by the displacement (Blue Curve), as shown in Figure 3. The system's dynamics and external disturbances initially cause the displacement to increase. However, the displacement gradually stabilizes close to the intended setpoint (1.0), demonstrating how effectively the feedback loop works to reduce oscillations. The rate at which the displacement changes over time is represented by the velocity (Orange Curve). The system's dynamics and control input cause the velocity to fluctuate. As the PID controller stabilizes the system, the velocity amplitude gradually decreases. The PID controller's corrective action to reduce the displacement error is reflected in the control input (Green Curve). The

initial spike demonstrates the controller's aggressive response to the significant initial error. The subsequent oscillations in the control input represent the adjustments made to stabilize the system. Over time, the feedback loop reduces oscillations by efficiently stabilizing the displacement. In this dynamic system, both velocity and control input behave as expected, with the control input continuously adjusting to minimize displacement errors.

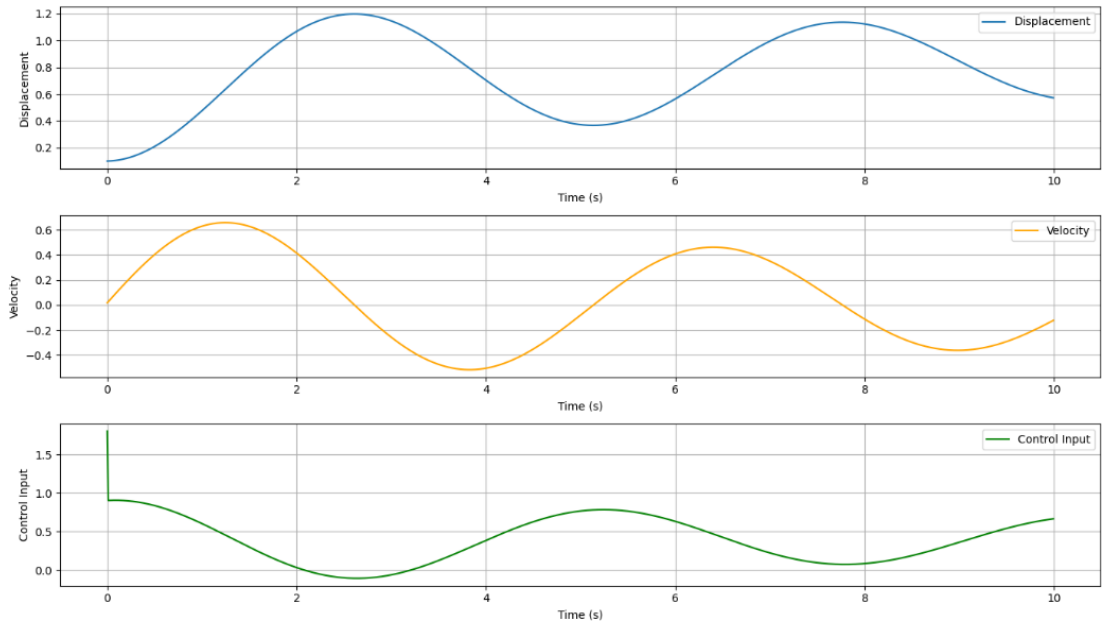

Figure-4 Dynamic Behavior of an Aeroservoelastic System with PID Feedback Control

Three subplots in the above figure 4 illustrate the dynamic response of an aeroservoelastic system under PID feedback loop control. First Plot: Time vs. Displacement The system's displacement with time, such as wing movement or system status, is shown by the blue curve. The system's inherent response and initial response to disturbances are reflected in the oscillatory pattern. The displacement gradually approaches the intended setpoint (stabilization), demonstrating the feedback controller's efficacy. The velocity (rate of change of displacement) is shown by the orange curve in the second plot (velocity vs. time). The system's reaction to outside factors and control activities causes velocity to fluctuate. As the system stabilizes, the oscillations' amplitude gradually drops, demonstrating the damping effect of the controller. The control input that the PID controller applies is shown by the green curve in the third plot (control input vs. time). The aggressive action made by the controller to lower the significant initial error is represented by the initial spike. As the system stabilizes, oscillations in the control input diminish with time. Over time, the PID controller successfully reduces displacement and velocity oscillations. Adaptive control inputs cause the system to stabilize around the target state, proving the effectiveness of the feedback loop. In order to counteract disruptions and stabilize the system, the control input dynamically adjusts.

Four subplots in the below figure 5 shows the dynamic response of an aeroservoelastic system under PID feedback controller management. Displacement vs. Time (Top Plot): The system's displacement (position) over time is shown by the blue curve. The system first oscillates as a result of natural dynamics and perturbations. By reducing oscillations over time, the PID controller stabilizes the displacement close to the intended setpoint. Velocity vs. Time (Second Plot): The velocity, or the rate at which displacement changes over time, is shown by the orange curve. The system's transient behavior causes the velocity to fluctuate at first, but as the controller stabilizes the system, the velocity steadily drops. The amplitude of oscillations decreases, indicating the damping effect of the system. Third Plot: Time vs. Control Input: The control input (corrective action or force) applied by the PID controller is displayed by the green curve. The early spike is a reflection of the aggressive response to a major initial error. The control input oscillates as the PID controller adjusts to stabilize the system; the magnitude of the oscillation progressively diminishes as the error reduces. Error vs. Time (Bottom Plot): the error (difference between the setpoint and the actual displacement) is shown by the red curve. As the PID controller modifies the control input to move the system closer to the intended state, the error gradually drops over time. The error's oscillations correspond to the displacement and velocity's transient reaction. Through gradual reduction of displacement and velocity oscillations, the PID feedback loop efficiently stabilizes the system. As the system gets closer to steady-state stability, the control input dynamically modifies to minimize error, with a decreasing intensity. The PID controller's ability to reach the intended setpoint is confirmed by the diminishing error curve.

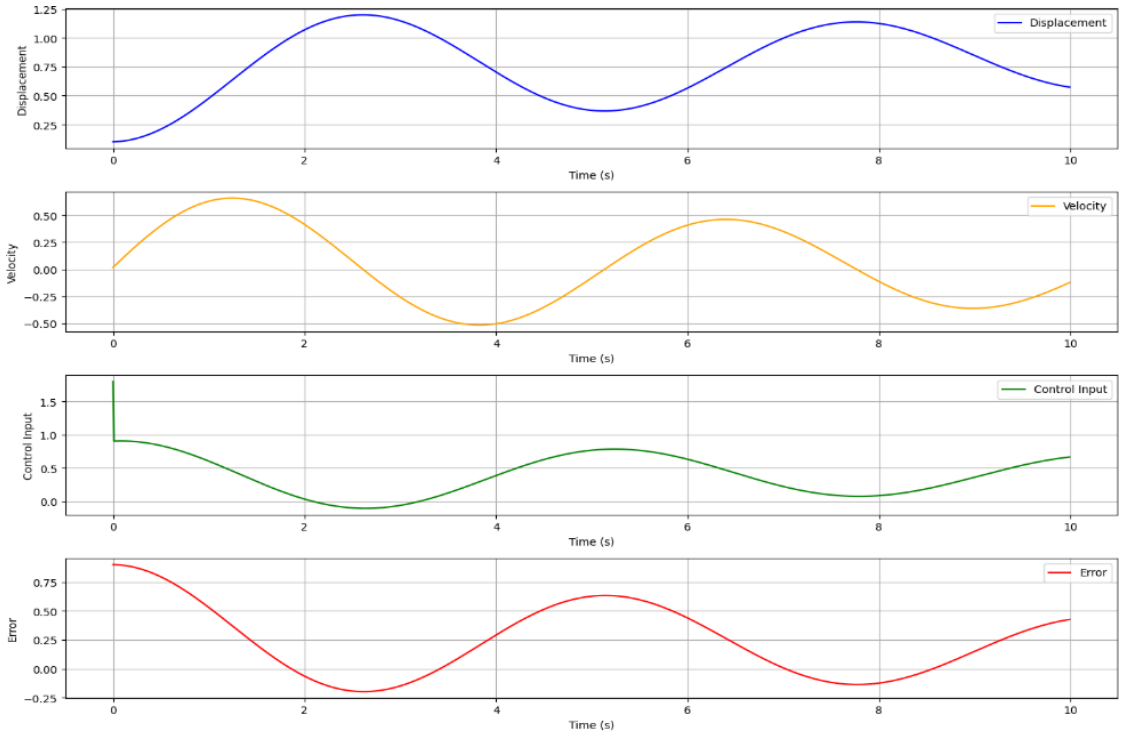

Figure-5 Dynamic Response and Error Reduction in an Aeroservoelastic System with PID Control

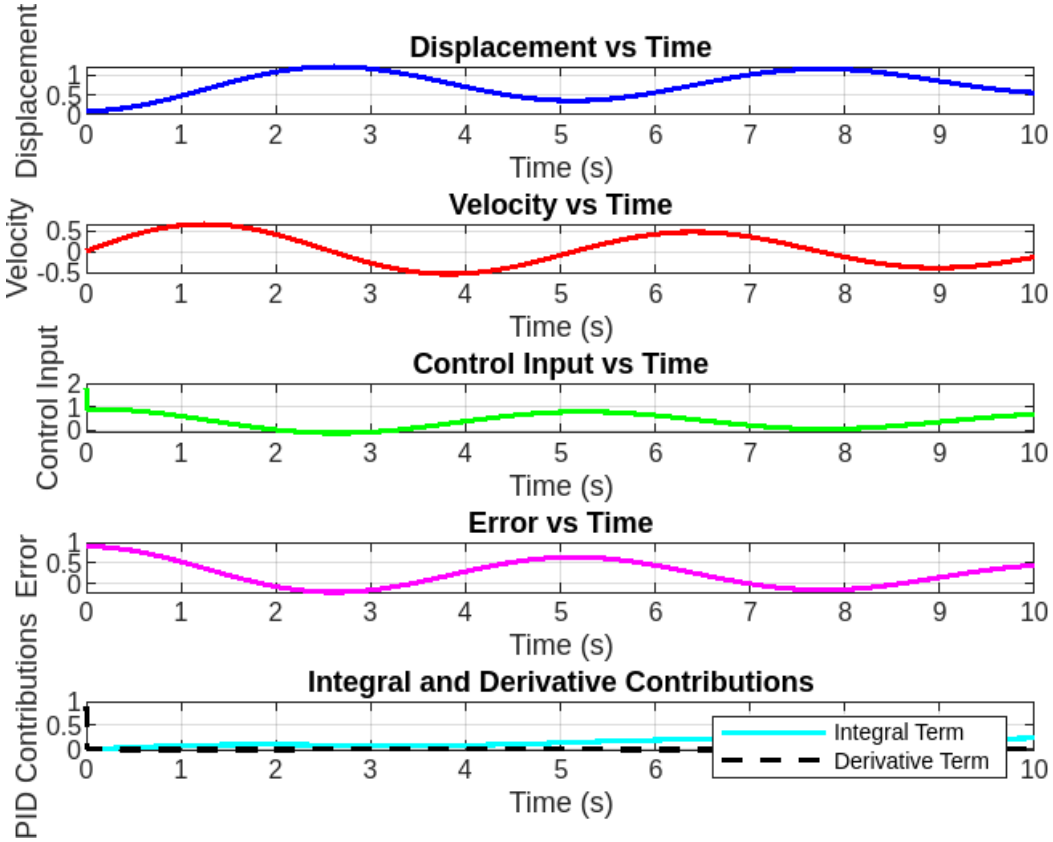

Figure-6 Dynamic Response and PID Contributions in an Aeroservoelastic System with Feedback Control

The dynamic behavior of an aeroservoelastic system under a PID feedback control mechanism is shown in in the figure 6, with several subplots elucidating various facets of the system's response, which emphasizes on a comprehensive visualization of the dynamic behavior of an aeroservoelastic system under a PID feedback control mechanism. Top Plot: Displacement vs. Time: The displacement of the system over time is shown by the blue curve. Initial conditions or outside perturbations cause the displacement to fluctuate. Displacement oscillations lessen and the system gets closer to the intended setpoint as the PID controller stabilizes it. Velocity vs. Time (Second Plot): The velocity, or rate at which displacement changes, is displayed over time by the red curve. The velocity curve, which shows the damping impact of the feedback loop, stabilizes as the control system lessens displacement oscillations, reflecting the system's transitory oscillatory response. Third Plot: Control Input vs. Time: The green curve shows the control input that the PID controller applies in order to reduce error. In reaction to a significant initial error, the controller's forceful corrective action is represented by the initial spike. As the system stabilizes, subsequent oscillations get softer and less severe. The error (difference between the desired setpoint and the actual displacement) is represented by the magenta curve in the fourth plot of error vs. time. The error is substantial at first, but it gradually gets smaller as the PID controller moves the system closer to the intended state. As stability improves, the error oscillations progressively decrease. Integral and Derivative Contributions (Bottom Plot): The accumulated error's contribution to the control input is displayed by the cyan line (Integral Term). It helps to get rid of steady-state mistakes by steadily rising and stabilizing. The contribution of the rate of change of error to the control input is shown by the dashed black line (derivative term). The PID controller stabilizes displacement and velocity while efficiently reducing error. The derivative term predicts changes and reduces oscillations, while the integral term addresses accumulated error and promotes long-term stability. Adaptability is demonstrated by the dynamic adjustment of control input based on system behavior.

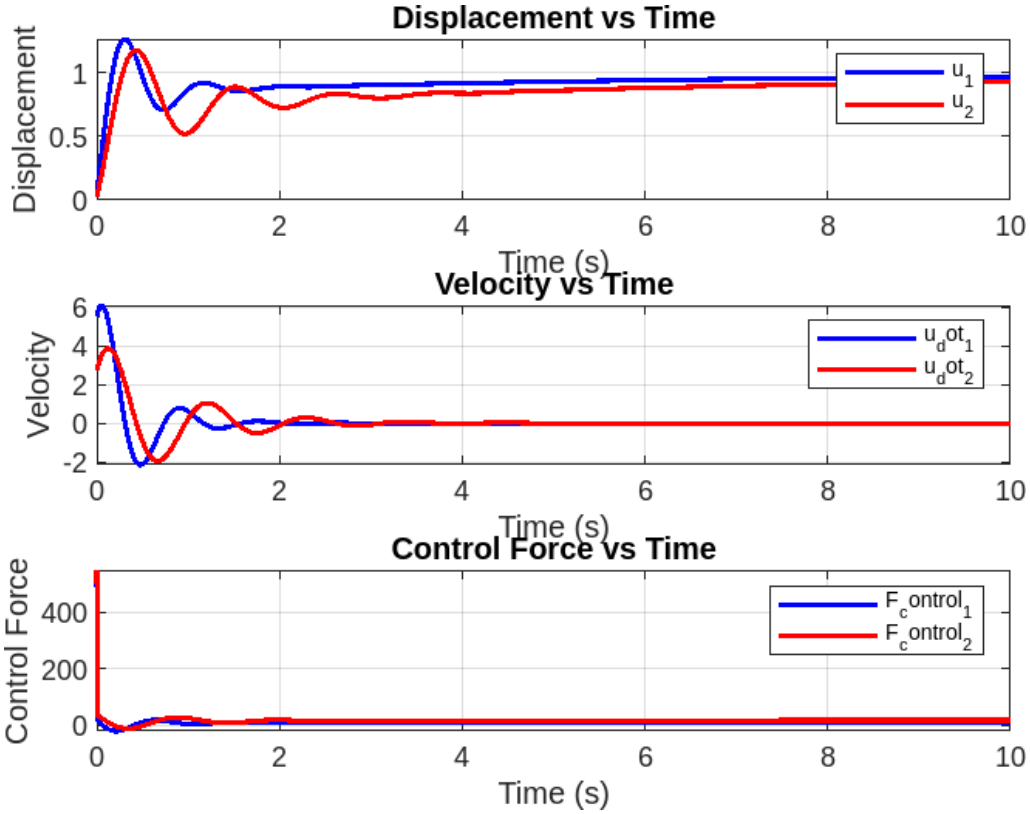

Figure-7: Dynamic Response and Control Force of Two Systems under Feedback Control

The above figure 7 consists of three subplots that two systems (u_1, u_2) behave while they are under feedback control. Top Plot: Displacement vs. Time: The first system's displacement over time is shown by the blue curve (u_1). The second system's displacement is shown by the red curve (u_2). Due to initial conditions or external disturbances, both systems exhibit initial oscillations. These oscillations are gradually reduced by the feedback control, and the displacements stabilize close to the intended setpoint. Velocity vs. Time (Middle Plot): The first system's velocity, or rate of change of displacement, is shown by the blue curve (u_{dot1}). The second system's velocity is shown by the red curve (u_{dot2}). When the system approaches steady- state stability, the velocity first shows brief fluctuations before steadily stabilizing to almost zero. Variations in the two systems' oscillatory behavior could be a sign of different system dynamics or beginning conditions. Control Force vs Time (Bottom Plot): The feedback controller's control force applied to the first system is shown by the blue curve ($\boldsymbol{F}_{control1}$). The control force applied to the second system is shown by the red curve ($\boldsymbol{F}_{control2}$). As the controller responds to the significant initial error, both systems show a strong initial control force (large spike). As the displacement and velocity errors in the system get closer to zero, the control force gradually drops and stabilizes. Both systems exhibit efficient feedback control stabilization. Over time, initial displacement and velocity oscillations are effectively dampened. In order to minimize errors, the control force dynamically modifies, beginning with a high intensity and decreasing as the system stabilizes. The responses of the two systems differ, indicating different dynamics or control parameters.

5. Conclusion

The current work addresses the problems caused by structural flexibility and aerodynamic forces by effectively implementing a feedback control system to stabilize an aeroelastic model of an aircraft wing. The system efficiently suppresses limit cycle oscillations (LCOs) and maintains stability by dynamically minimizing displacement and velocity errors through the combination of state-space dynamics and a PID controller. The findings show that the derivative component predicts and reduces oscillatory behavior, the integral component eliminates steady-state errors, and the proportional component quickly corrects displacement deviations. The efficiency of PID feedback in managing intricate aeroelastic interactions is demonstrated by this synergistic control, which dramatically reduces oscillations and stabilizes the system around the intended setpoint. The importance of feedback systems in contemporary aerospace applications is highlighted by this work, which ensures structural integrity, performance optimization, and safety in dynamic environments. Aeroelastic systems can be stabilized by developing a robust feedback control system using MATLAB and Industry 4.0 concepts. In aircraft applications, the integration of digital twins, dynamic feedback, and real-time monitoring enhances system performance and ensures safety.

6. Acknowledgement

The authors would like to express their gratitude to Dayananda Sagar University and Bosch Rexroth for facilitating support to carry out this research work.

7. Conflict of Interest

The author declares no competing conflict of interest.

8. Funding Information

No funding was received to support this study.

9. Data Availability Statement

No data were used or taken from any external source for this research. All the findings and analyses presented in this study are based on the author's original work.

10. References

[1] Chinesta, F., Cueto, E., Abisset-Chavanne, E., Duval, J. L., & El Khaldi, F. (2020). Virtual, digital and hybrid twins: A new paradigm in data-based engineering and engineered data. *Archives of Computational Methods in Engineering*, *27*(1), 105–134. https://doi.org/10.1007/s11831-018-9301-4.

[2] Singh, S., Shehab, E., Higgins, N., Fowler, K., Tomiyama, T., & Fowler, C. (2018). Challenges of digital twin in high value manufacturing. *AE Technical Papers*. https://doi.org/10.4271/2018-01-1928.

[3] Grieves, M., & Vickers, J. (2016). Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems. In *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches* (pp. 85–113). Springer International Publishing. https://doi.org/10.1007/978-3-319-38756-7_4.

[4] Li, L., Aslam, S., Wileman, A., & Perinpanayagam, S. (2022). Digital twin in aerospace industry: A gentle introduction. *IEEE Access*, *10*, 9543–9562. https://doi.org/10.1109/ACCESS.2021.3136458.

[5] Boschert, S., & Rosen, R. (2016). Digital twin—the simulation aspect. In *Mechatronic Futures: Challenges and Solutions for Mechatronic Systems and Their Designers* (pp. 59–74). Springer International Publishing. https://doi.org/10.1007/978-3-319-32156-1_5.

[6] Tao, F., Cheng, J., Qi, Q., Zhang, M., Zhang, H., & Sui, F. (2018). Digital twin-driven product design, manufacturing and service with big data. *International Journal of Advanced Manufacturing Technology*, *94*(9–12), 3563–3576. https://doi.org/10.1007/s00170-017-0233-1.

[7] Hazbon Alvarez, O., Gutierrez Zea, L., Bil, C., Napolitano, M., & Fravolini, M. L. (2019). Digital twin concept for aircraft sensor failure. In *Advances in Transdisciplinary Engineering* (pp. 370–379). IOS Press BV. https://doi.org/10.3233/ATDE190143.

[8] Brunton, S. L., et al. (2021). Data-driven aerospace engineering: Reframing the industry with machine learning. *AIAA Journal*, *59*(8), 2820–2847. https://doi.org/10.2514/1.J060131.

[9] Idoko, P., Ezeamii, G. C., Idogho, C., Peter, E., Obot, U. S., & Iguoba, V. A. (2024). Mathematical modeling and simulations using software like MATLAB, COMSOL and Python. *Magna Scientia Advanced Research and Reviews*, *12*(2), 062–095. https://doi.org/10.30574/msarr.2024.12.2.0181.

[10] Salvy, P., Fengos, G., Ataman, M., Pathier, T., Soh, K. C., & Hatzimanikatis, V. (2019). PyTFA and matTFA: A Python package and a MATLAB toolbox for thermodynamics-based flux analysis. *Bioinformatics*, *35*(1), 167–169. https://doi.org/10.1093/bioinformatics/bty499.

[11] Raja, S. V., Vishwa, A., India, V., & Unnikrishnan, A. (n.d.). Moving towards Industry 4.0: A systematic review. *International Journal of Pure and Applied Mathematics.* Retrieved from http://www.ijpam.eu

[12] Mohamed, M. (2018). Challenges and benefits of Industry 4.0: An overview. *International Journal of Supply and Operations Management*, *5*(3), 256–265. Retrieved from www.ijsom.com

[13] Da Xu, L., Xu, E. L., & Li, L. (2018). Industry 4.0: State of the art and future trends. *International Journal of Production Research*, *56*(8), 2941–2962. https://doi.org/10.1080/00207543.2018.1444806.

[14] Roblek, V., Meško, M., & Krapež, A. (2016). A complex view of Industry 4.0. *Sage Open*, *6*(2). https://doi.org/10.1177/2158244016653987.

[15] Yang, F., & Gu, S. (2021). Industry 4.0, a revolution that requires technology and national strategies. *Complex and Intelligent Systems*, *7*(3), 1311–1325. https://doi.org/10.1007/s40747-020-00267-9.

[16] Vaidya, S., Ambad, P., & Bhosle, S. (2018). Industry 4.0 – A glimpse. *Procedia Manufacturing*, *20*, 233–238. https://doi.org/10.1016/J.PROMFG.2018.02.034.

[17] Peres, R. S., Jia, X., Lee, J., Sun, K., Colombo, A. W., & Barata, J. (2020). Industrial artificial intelligence in Industry 4.0—Systematic review, challenges and outlook. *IEEE Access*. https://doi.org/10.1109/ACCESS.2020.3042874.

[18] Castro, H., Costa, F., Ferreira, L., Avila, P., Putnik, G. D., & Cruz-Cunha, M. (2022). Data science for Industry 4.0: A literature review on open design approach. *Procedia Computer Science*, *202*, 877–884. https://doi.org/10.1016/j.procs.2022.08.106.

[19] Sharabov, M., & Tsochev, G. (2020). The use of artificial intelligence in Industry 4.0. *Problems of Engineering Cybernetics and Robotics*, *73*. https://doi.org/10.7546/pecr.73.20.02.

[20] Yang, C., Zhengjie, W., Meifang, G., & Yuanhang, W. (n.d.). Dynamics modeling and simulation of small flexible wing aircraft. https://doi.org/10.1109/ICMIC.2014.7020771.

[21] Svoboda, F., Hromcik, M., & Sika, Z. (2021). Adaptive control design for the aeroelastic wing. In *Proceedings of the 2021 23rd International Conference on Process Control, PC 2021* (pp. 225–228). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/PC52310.2021.9447451.

[22] Smain, D., & Brahim, B. (2010). Modeling and control of flexible wing with trailing and leading edge control surfaces. In *Proceedings - UKSim 4th European Modelling Symposium on Computer Modelling and Simulation, EMS2010* (pp. 254–259). https://doi.org/10.1109/EMS.2010.48.

[23] Botez, R., Cotoi, I., Doin, A., & Biskri, D. (2002). Method validation for aeroservoelastic analysis. In *Collection of Technical Papers - AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference* (pp. 2238–2248). https://doi.org/10.2514/6.2002-1481.

[24] Wüstenhagen, M., Kier, T., Pusch, M., Ossmann, D., Muhammad, Y. M., & Hermanutz, A. (2018). Aeroservoelastic modeling and analysis of a highly flexible flutter demonstrator. In *2018 Atmospheric Flight Mechanics Conference* (AIAA 2018-3150). https://doi.org/10.2514/6.2018-3150.

[25] Mcgowan, A.-M. et al. (n.d.). Aeroservoelastic and structural dynamics research on smart structures conducted at NASA Langley Research Center. Retrieved from https://ntrs.nasa.gov/citations/20040110282.

[26] Haghighat, S., Martins, J. R. R. A., & Liu, H. H. T. (n.d.). Aeroservoelastic design optimization of a flexible wing. https://doi.org/10.2514/1.C031344.

[27] Liu, Y., & Xie, C. (2018). Aeroservoelastic stability analysis for flexible aircraft based on a nonlinear coupled dynamic model. *Chinese Journal of Aeronautics*, *31*(12), 2185–2198. https://doi.org/10.1016/j.cja.2018.08.019.

[28] Lacabanne, M., & Laporte, A. (n.d.). ICAS 2000 CONGRESS 471.1 Progress in the prediction of aeroservoelastic instabilities on large civil transport aircraft.

[29] Balas, G. J., & Seiler, P. J. (n.d.). Robust flutter analysis for aeroservoelastic systems.

[30] Dowell, E. H. (n.d.). *Solid mechanics and its applications: A modern course in aeroelasticity (5th ed.)*. Retrieved from http://www.springer.com/series/6557.