Development of a Software Tool for Learning the Fundamentals of CubeSat Angular Motion

Victor Romero-Alva[1], Angelo Espinoza-Valles[2]
Faculty of Engineering, Universidad Tecnológica del Perú, Peru[1]
Inter-University Department of Space Research, Samara National Research University, Russian Federation[2]

***Abstract*—The development of tools for understanding and simulating CubeSat angular motion is essential for both educational and research purposes in space technology. In this context, this paper presents the development of a MATLAB-based software tool designed to facilitate the comprehension of CubeSat angular motion. This tool allows users to simulate CubeSat dynamics by adjusting parameters, such as initial conditions and physical properties, enabling the observation of different types of motion, including rotatory, oscillatory, both stable and unstable behaviors. The mathematical models selected for simulating the CubeSat dynamics are presented. The interface of the tool, designed for intuitive parameter input and visualization of phase portraits of the system under consideration, is described. The software is demonstrated using a CubeSat 3U configuration, and simulation results, including angle of attack, angular velocity, and altitude decay, are presented. This tool aims to enhance the understanding of CubeSat angular motion, contributing to the design and operation of CubeSat missions in low Earth orbit.**

Keywords—CubeSat; angular motion; simulation; learning tool MATLAB

I. INTRODUCTION

CubeSats, a class of nanosatellites, have become popular in educational and commercial activities in space due to their cost effectiveness and versatility [1, 2]. The proliferation of CubeSats, typically deployed in low Earth orbit (LEO), has transformed space exploration and satellite-based research. Previously, solutions to many complex space problems could be imagined possible only with the use of larger spacecraft [3-5].

One critical aspect of their operation is angular motion, which determines orientation of the spacecraft in space. Understanding how a CubeSat experiences angular motion and how to control its attitude is vital for mission success, especially for applications involving imaging, communication, and scientific measurements [6, 7]. Despite their relatively small dimensions, CubeSats adhere to the same fundamental principles of orbital mechanics and attitude control as larger spacecraft. These principles are influenced by external forces such as gravitational and aerodynamic torques, which can vary based on the altitude, mass distribution, and geometric properties of the satellite [8].

In practice, achieving stable or predictable attitude control is essential for both passive and active stabilization methods and requires a firm comprehension of how these forces are related to the physical characteristics of the CubeSat. Given the increasing complexity of CubeSat missions, there is a growing need for intuitive and interactive learning tools that simplify the study of its dynamics.

Traditional methods of teaching these concepts often rely on theoretical mathematical models, which can be difficult for learners to fully comprehend without visual aids or practical experimentation. Consequently, in recent years, the development of graphical user interfaces as learning tools has gained traction across multidisciplinary fields, demonstrating their potential to facilitate complex subject matter through interactive and intuitive designs.

For instance, Botha and Marais developed a GUI to support learning in artificial intelligence, demonstrating the effectiveness of visual and interactive tools in simplifying sophisticated AI concepts [9]. Similarly, Mohd and Hashim introduced a deep learning interface designed to assess fruit quality, which exemplifies how such interfaces can aid in applied fields like food engineering [10]. These methodologies are also employed in chemometrics, as seen in the work of Chiappini et al., who developed a MATLAB GUI for multivariate calibration [11]. Gasparic et al. developed a GUI that enhances integrated development environment usability by providing command recommendations, demonstrating through user studies that such interfaces are essential for effective developer support [12]. Mishra et al. [13] developed a MATLAB-based interface that simplifies multi-block data analysis by offering integrated visualization, classification, and pre-processing capabilities, thereby making advanced chemometric methods accessible to users in industrial applications. Victoria et al. introduced a GUI for topology design in engineering structures, which underscores the broad applicability of such interfaces in engineering design processes [14]. Furthermore, GUIs have been implemented in astronomy, for example, Errazzouki et al. created a MATLAB interface to facilitate the acquisition of single star SCIDAR data, facilitating data processing [15].

These developments collectively underscore the effectiveness of interactive graphical interfaces in making complex theoretical concepts more accessible and engaging. However, despite these successful implementations, there remains a distinct need for a simplified simulation tool that provides relatively accurate representations of CubeSat dynamics while remaining accessible to students and newcomer CubeSat developers. Thus, the objective of this paper is to present a software learning tool designed to simulate the CubeSat angular motion through an intuitive interface, while facilitating an easy understanding of CubeSat angular motion fundamentals. This MATLAB-based tool allows users to adjust key design parameters, observe different types of motion, and visualize phase portraits of the system, providing valuable insights into CubeSat dynamics.

The remainder of the article is organized as follows: Section II reviews related work in tools for simulating spacecraft dynamics. Section III presents the theoretical concepts underlying CubeSat angular motion and describes the mathematical models implemented in the tool, with emphasis on the key assumptions made for model simplification. Section IV presents the results, demonstrating the utility of the tool through a practical example based on a CubeSat 3U configuration. Section V provides a discussion of the results, highlighting the limitations, and potential areas for future work. Finally, Section VI presents the conclusions, summarizing the key contributions and significance of the developed tool."

II. RELATED WORK

In the field of spacecraft dynamics, several sophisticated software tools and packages have been developed to simulate both angular and orbital motion of spacecraft. For instance, Hadi and Sasongko developed a CubeSat design and visualization tool in MATLAB that not only calculates key satellite parameters but also simulates dynamic responses, thereby supporting the analysis and design of CubeSat systems [16].

Ivanov et al. introduced a software package capable of simulating passive and controlled angular and orbital motion of near-Earth satellites [17]. While highly versatile, the reliance on specific mathematical models can make it less accessible for users seeking a simpler introduction to CubeSat dynamics. Moreover, as the software is developed in C++, it may be more difficult to modify or add new modules compared to more user-friendly environments like MATLAB.

Ezzat et al. provided a tool for satellite orbit tracking based on the Keplerian system. Although this method demonstrates high precision and robust visualization for orbit tracking, it does not simulate the angular motion relative to the spacecraft's center of mass. This omission is significant because understanding the angular dynamics is critical for evaluating the attitude stability and control of the spacecraft, which are essential for mission success [18].

Similarly, Turner developed an open-source spacecraft simulation and control software based on an application programming interface intended for both research and educational purposes [19]. This framework delivers high-fidelity simulation routines and efficient, fast-running code, making it a powerful tool for detailed analysis. However, its effective use requires a robust understanding of programming, which may present a barrier for users with limited technical expertise.

Shirobokov and Trofimov developed the KIAM Astrodynamics Toolbox, a robust software library for spacecraft orbital motion that incorporates Fortran modules for implementing astrodynamical functions and Python modules for their compilation [20]. While the toolbox is effective for detailed simulations and ensures computational speed, its primary focus on orbital dynamics may overlook key aspects of CubeSat attitude control and angular motion. While this advanced tool is effective, it may not be suitable for educational purposes. The reliance on a traditional programming language, as Fortran, alongside Python interfaces, can pose accessibility challenges for users without a solid programming background.

While the reviewed tools offer detailed and accurate simulations suitable for advanced research, there is a noticeable gap when it comes to introductory educational applications. Many of these tools are designed with high-fidelity modeling in mind, which can sometimes render them less accessible for beginners or users without extensive programming experience. In summary, the literature reveals two key trends in the development of educational simulation tools: the integration of graphical user interfaces to help demystify theoretical concepts and the growing use of open-source, extensible software frameworks that allow for both accurate and customizable simulations.

III. THEORY OF CUBESAT ANGULAR MOTION

In this section, we provide an overview of the theoretical concepts and mathematical models, which underlie CubeSat angular motion, focusing on dynamic equations generally used in CubeSat missions.

A. Simplifications and Assumptions for CubeSat Dynamics Modeling

In the development of this interactive learning tool for CubeSat angular motion, several foundational assumptions are made to simplify the complex dynamics involved and facilitate an effective educational framework.

A primary assumption is that the CubeSat exhibits planar angular motion in a circular orbit, subject to the influence of gravitational and aerodynamic torques. This assumption enables a focused analysis of the rotational behavior by limiting the scope to the two principal forces acting on the CubeSat. This assumption is particularly appropriate for CubeSats operating in LEO, allowing the model to provide a simplified and close representation of their attitude dynamics.

Given that the geometry of a CubeSat is generally a parallelepiped, it can be considered a dynamically symmetric body. Therefore, its moments of inertia about the *y*- and *z*-axes are assumed to be equal ($I = I_y = I_z$). Furthermore, for form factors larger than 1U, the longitudinal moment of inertia I_x is associated with the *x*-axis, which is the axis of symmetry and corresponds to the longest side of the CubeSat.

Due to the typical vertical stacking of components in CubeSats, it is assumed that the center of mass may be displaced from the center of pressure, primarily along the longitudinal axis, denoted as Δx. This is because the arrangement of components is more likely to create asymmetry along this axis. In contrast, displacements along the lateral *y*-axis and *z*-axis are considered negligible or zero. This assumption simplifies the analysis, particularly in assessing the impact of aerodynamic forces.

These simplifications are essential for creating a simulation that is both computationally manageable and pedagogically valuable for understanding the principles of angular motion in CubeSats.

B. Mathematical Model for CubeSat Angular Motion Modeling

In this study, the motion of a CubeSat relative to its center of mass is examined and its implemented based on the theorem of conservation of angular momentum.

According to study [21], an approximate model for the angular motion of a dynamically symmetric CubeSat in a circular orbit can be obtained by the following the expression:

$$\ddot{\alpha} - \frac{M_{ay}}{I}\sin\alpha - \frac{M_{gy}}{I}\sin\alpha = 0, \tag{1}$$

where, α is the angle of attack, M_{ay} is the aerodynamic moment, M_{gy} is the gravitational moment and I is the CubeSat transverse moment of inertia.

At lower altitudes, aerodynamic forces play a significant role in the dynamics of the CubeSat, while at higher altitudes, the influence of aerodynamic moments diminishes, and gravitational moments become more predominant. Despite the weakening of gravitational force with increasing altitude, its effects remain relevant and must be thoroughly analyzed to ensure effective control and stabilization. In this context, the gravitational moment is calculated using the following formula:

$$M_{gy} = -\frac{3\mu}{2r^3}(I_x - I)\sin 2\alpha \tag{2}$$

where, μ is the Earth's gravitational parameter, r is the CubeSat position radius-vector, I_x is the CubeSat longitudinal moment of inertia.

For CubeSats, the angular acceleration caused by aerodynamic moments is significantly higher. These moments have a much greater influence on smaller CubeSats than on larger, more massive ones, even when both have the same relative center of mass displacement and volumetric density [21]. The moment of aerodynamic drag force is calculated as follows:

$$M_{ay} = -c_0 S \Delta\bar{x} q l \sin\alpha \tag{3}$$

where $c_0 = 2.2$ is the drag coefficient, S is the characteristic area of the CubeSat perpendicular to the flight velocity vector, $\Delta\bar{x} = \Delta x / l$ is the relative distance from the center of pressure to the center of mass along the x axis of the CubeSat, q is the velocity pressure, l is the characteristic length of the CubeSat.

In the previous model, the orbital dynamics of the CubeSat were heavily dependent on the flight altitude, as this parameter directly influences atmospheric drag and other perturbative forces. Given the significance of altitude in attitude, we aim to utilize the approximate formula to predict altitude variations over time. This allows for a close estimation of the orbital evolution, especially in LEOs where altitude plays a critical role in its long-term stability and performance. For CubeSats, the ballistic coefficient tends to be higher compared to larger satellites, which leads to a shorter orbital lifetime due to the increased drag forces acting upon them at a given altitude. Under the assumption that the descent angle of a nanosatellite in a nearly circular orbit remains small and varies slowly, the altitude change over time can be estimated using the equation [22]:

$$\dot{h} = -\frac{2\sigma q V}{g} \tag{4}$$

where σ is the ballistic coefficient, V is the flight velocity and g is the Earth's gravity at a specific altitude.

Assuming the flow is free molecular and the impact of gas molecules is completely inelastic, the ballistic coefficient of a CubeSat can be determined by the following formula [8]:

$$\sigma = -\frac{c_0 S}{m} \tag{5}$$

where m is the CubeSat mass.

IV. RESULTS

In this section the developed software tool, the CubeSat Angular Motion Simulator, is introduced.

A. Learning Tool Interface Overview

The proposed interface was implemented and verified using MATLAB programming language. This tool provides an interactive environment where users can explore the dynamic behavior of CubeSats by modifying various physical and initial conditions of the angular motion. The primary features include real-time visualization of angular motion, user input of critical parameters, and the generation of phase portraits, allowing for an in-depth exploration of both theoretical and practical aspects of CubeSat dynamics.

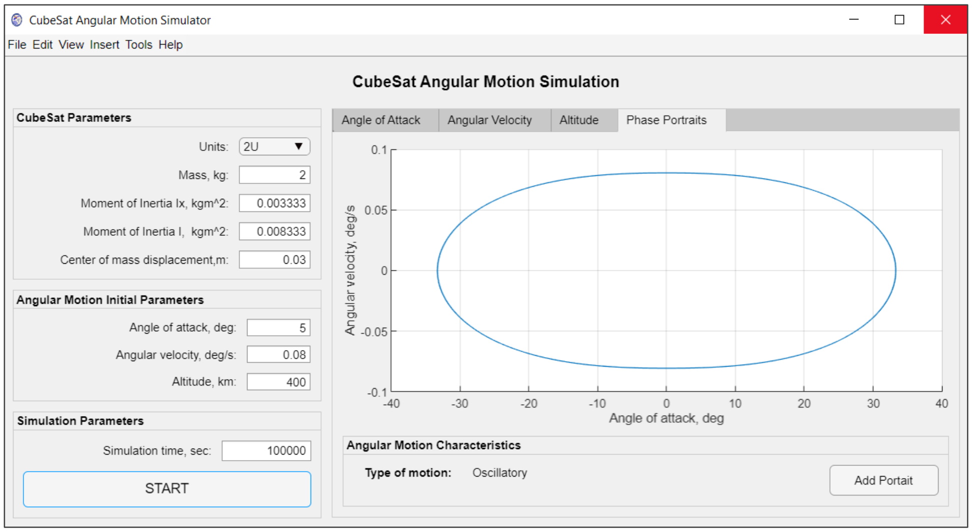
The interface of the CubeSat Angular Motion Simulator is designed for ease of use, as shown in Fig. 1. It is divided into several distinct panels: one for inputting CubeSat parameters, another for initializing angular motion conditions, a third for adjusting simulation time, and finally, a visualization panel that plots the phase portraits of angular motion.

B. Initial Data for Simulation

The input data required for the simulation includes a range of essential parameters that define the dynamic characteristics of the CubeSat. Specifically, it includes the geometric properties, center-of-mass configuration, inertial characteristics, and aerodynamic features of the nanosatellite. Additionally, the simulation requires the altitude of the CubeSat during flight and the initial conditions of its angular motion. Collectively, these parameters are crucial for accurately modeling the angular dynamics of the nanosatellite and facilitating a comprehensive understanding of its behavior in space.

The leftmost panel, labeled “CubeSat Parameters”, allows users to define the physical characteristics of the CubeSat. These include:

- Units: The tool supports various CubeSat form factor, such as 1U up to 12U. Users can select the required size from a dropdown menu.
- Mass: The mass of the CubeSat in kilograms.
- Moments of inertia: For determining the resistance to rotational motion users input the longitudinal moment of inertia I_x and the transversal moment of inertia I, both in kg·m².



CubeSat Angular Motion Simulation

CubeSat Parameters

Units: 2U
Mass, kg: 2
Moment of Inertia Ix, kgm^2: 0.003333
Moment of Inertia I, kgm^2: 0.008333
Center of mass displacement,m: 0.03

Angular Motion Initial Parameters

Angle of attack, deg: 5
Angular velocity, deg/s: 0.08
Altitude, km: 400

Simulation Parameters

Simulation time, sec: 100000

Angular Motion Characteristics

Type of motion: Oscillatory

Fig. 1. Main interface of the CubeSat angular motion software.

- Center of mass displacement: A key feature of the tool is the ability to account for displacement of the center of mass along the longitudinal axis, which is input in meters.

By adjusting these inputs, users can simulate various CubeSat configurations, from simple single unit designs to more complex with multiple units.

Below the CubeSat Parameters section, users define the initial conditions of angular motion in the “Angular Motion Initial Parameters” panel.

- Angle of attack α: This parameter, expressed in degrees, defines the initial angle between the CubeSat longitudinal axis and the flight velocity vector.
- Angular velocity $\dot{\alpha}$: The initial angular velocity, given in degrees per second, specifies the rate of rotation of the angle of attack.
- Altitude: The orbital altitude of the CubeSat in kilometers.

The Simulation Parameters panel allows users to specify the total simulation time in seconds, enabling them to explore short-term or long-term behavior of the angular motion.

C. Phase Portraits Section

One of the features of the tool is the capability to generate phase portraits. A phase portrait represents the trajectory of the angular motion in phase space, where the angle of attack is plotted against angular velocity. This visual representation allows users to easily identify whether the motion is periodic, stable, or exhibits any chaotic tendencies. The phase portraits tab provides users with the option to save and compare multiple simulations, offering an understanding of how variations in initial conditions or CubeSat parameters impact the system. By selecting the “Add Portrait” button, users can overlay results from different simulation runs, facilitating comparative analysis.

For example, the phase portrait presented in Fig. 2 illustrates an analysis of angular motion in a planar representation, generated using the software tool. This example incorporates five distinct types of motion added using the “Add Portrait” button. The plot offers an insightful look into how different types of motion, rotatory and oscillatory, can be represented, along with the critical role of the separatrix in dividing the regions of possible motion.

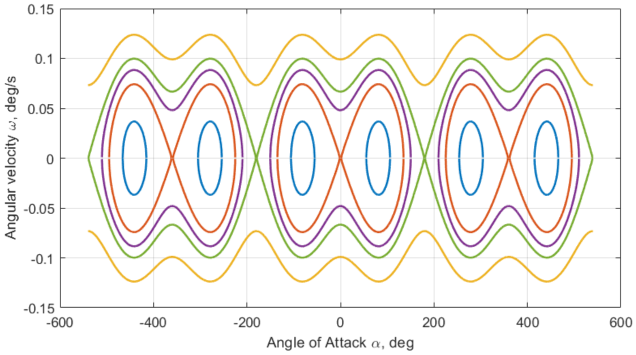


Fig. 2. Opportunity for study of phase portraits.

The phase portrait maps the relationship between the angle of attack α on the x-axis, and the angular velocity ω on the y-axis. Each trajectory corresponds to a specific type of motion, representing how the angular velocity changes as the angle of attack evolves.

1) Rotatory motion: Represented by the yellow curve, the rotatory motion is characterized by the CubeSat spinning continuously, with no return to its original orientation.

2) Oscillatory motion: In contrast, oscillatory motion is represented by closed trajectories (blue and purple curves). This type of motion implies that the angular velocity periodically reverses direction, leading to a back-and-forth movement around a stable point or equilibrium. Physically, this corresponds to a situation where the CubeSat is wobbling or oscillating without completing a full rotation.

3) Separatrix: The green and red curves are separatrices that delineate the boundary between oscillatory and rotatory motion. This curve indicates the critical point at which a small change in angular velocity or angle of attack can result in a significant shift from oscillatory to rotatory motion.

A more detailed analysis of determination of equilibrium positions for CubeSats can be found in the work [23].

Finally, below the graph, the software provides a real-time classification of the angular motion of the CubeSat under the label "Angular Motion Characteristics". Depending on the initial conditions and the physical parameters, the tool identifies the motion as oscillatory or rotary.

D. Data Output and Visualization

Once the parameters described in section B are set, the user can start the simulation by pressing the START button. The software tool provides a range of outputs that allow users to analyze the CubeSat dynamics over time. As an illustrative example, a simulation was conducted using the parameters presented in Tables I and II, with a simulation time of 100,000 seconds.

Table I details the design characteristics for a CubeSat 3U, while Table II outlines the initial angular motion parameters used for the simulation.

TABLE I. CUBESAT SIMULATION PARAMETERS

Parameter	Value
Mass, kg	3
Longitude, m	0.3
Moment of Inertia I_x, kg·m²	0.005
Moment of Inertia I, kg·m²	0.025
Center of mass displacement Δx, m	0.03

TABLE II. INITIAL ANGULAR MOTION PARAMETERS

Parameter	Value
Angle of attack, deg	45
Angular velocity, deg/s	0.15
Altitude, km	400

Fig. 3 shows the variation of the angle of attack over time, where a clear oscillatory behavior is observed. This periodic oscillation suggests that the orientation of the CubeSat relative to its orbit changes continuously due to perturbation torques.

Analogically, Fig. 4 presents the variation of the angular velocity over time.

The phase portrait shown in Fig. 5 plots angular velocity against the angle of attack. The curve represents oscillatory motion around the equilibrium position, where $\alpha = 0$.

Finally, Fig. 6 depicts the variation in flight altitude over time for different CubeSat with form factor 2U and 3U. This plot demonstrates the gradual decay in altitude, which is typical for satellites in low Earth orbit due to atmospheric drag. Also, this graphic demonstrates that the CubeSat 2U exhibits a faster loss of altitude compared to the CubeSat 3U.

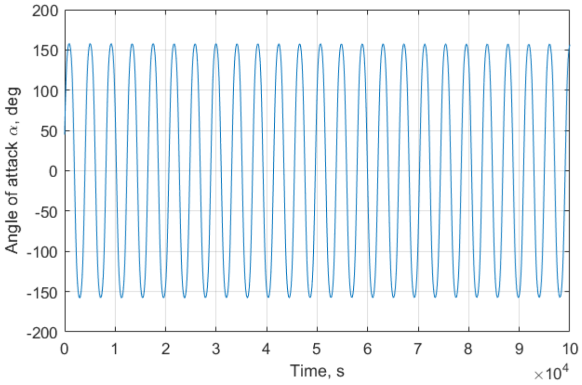


Fig. 3. Variation of the angle of attack during simulation.

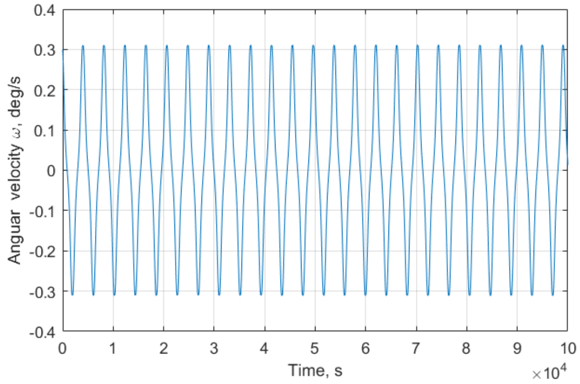


Fig. 4. Variation of the angular velocity during simulation.

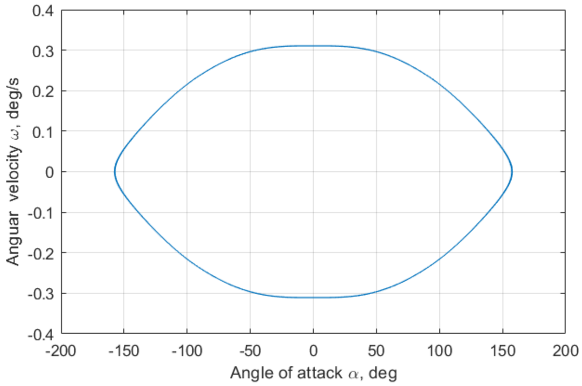


Fig. 5. Phase portrait during simulation.

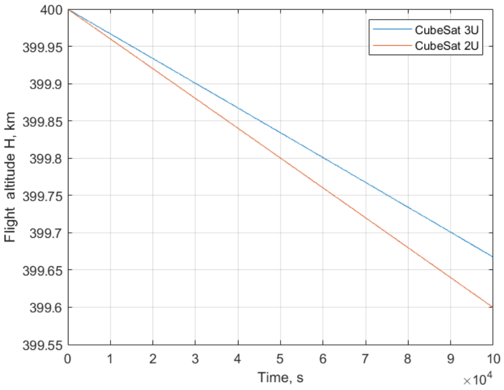


Fig. 6. Flight altitude during simulation for different CubeSats.

E. Validation and Verification

The validation and verification of the developed software tool were conducted through a combination of analytical benchmarking, sensitivity analysis, and software testing methodologies to ensure both numerical accuracy and computational reliability.

To validate the accuracy of the numerical simulations, benchmark comparisons were performed against analytical solutions derived from rigid-body dynamics. The presented mathematical models for angular motion serve as the theoretical reference for evaluating the correctness of the implemented models. For specific cases where closed-form solutions exist, numerical integration results were shown to closely replicate the expected angular velocity trajectories and phase portraits. Additionally, limiting cases were analyzed, including scenarios where aerodynamic torques were negligible, allowing direct verification against classical torque-free rotational motion solutions. The capacity of the software to capture equilibrium conditions and the expected transitions between oscillatory and rotatory regimes further reinforced the validity of the underlying mathematical framework.

Beyond validation with theoretical models, a sensitivity analysis was conducted to assess the robustness of the software in response to variations in physical parameters. Perturbations in the transverse and longitudinal moments of inertia were introduced to evaluate their effects on rotational stability, confirming that small deviations in these properties resulted in expected dynamic shifts consistent with established CubeSat dynamics theory. The center-of-mass displacement along the longitudinal axis was systematically varied to examine its impact on aerodynamic torques, demonstrating the ability to predict stability conditions influenced by structural asymmetry. Additionally, the effects on altitude were examined, confirming that at lower altitudes, increased atmospheric drag induced more pronounced oscillatory damping, while at higher altitudes, gravitational torques became the dominant perturbation force. These results closely aligned with theoretical expectations and prior research on CubeSat aerodynamics.

In addition to physical validation, software verification techniques were applied to ensure the integrity of the computational implementation. A unit testing framework was integrated within the MATLAB environment, employing modular tests for key computational functions, including numerical integration routines, moment calculations, and phase portrait generation. Each component was individually validated against known analytical solutions or benchmark datasets to detect discrepancies. Furthermore, consistency checks were performed by systematically varying input parameters and ensuring that the outputs adhered to expected trends. Edge-case testing was also conducted to evaluate system behavior under extreme parameter values, ensuring numerical stability and robustness.

V. DISCUSSION

The results obtained from the developed CubeSat Angular Motion Simulator demonstrate its effectiveness in visualizing and analyzing the angular motion dynamics of CubeSats in low Earth orbit. The phase portraits generated by the tool confirm the expected motion behaviors, distinguishing between oscillatory and rotatory regimes based on initial conditions. The clear identification of separatrices highlights the critical transition points between these regimes, providing valuable insights for both educational and preliminary design applications.

From an educational standpoint, the simulator bridges the gap between theoretical learning and practical application, raising critical questions about how simulation-based learning affects conceptual understanding of real-world physical phenomena. The reliance on digital models may shape how students perceive and interact with physical reality, potentially impacting their approach to problem-solving and experimental validation in aerospace engineering. By providing real-time visualization and interactive parameter adjustments, the tool enhances comprehension in aerospace engineering curricula, where abstract mathematical models often present significant learning challenges.

Despite its strengths, the current version of the tool has certain limitations. The assumption of planar angular motion simplifies the analysis but excludes three-dimensional effects, which may become significant in more complex mission scenarios. Additionally, the exclusion of external perturbations such as geomagnetic torques and solar radiation pressure limits the applicability of the model for higher-fidelity mission simulations. Future work should focus on extending the model to include these effects, as well as incorporating real CubeSat telemetry data for validation. Enhancing the tool with additional control system simulations could further increase its relevance for practical mission planning and research on space dynamics.

VI. CONCLUSION

This work introduced an interactive software tool designed to simulate and visualize CubeSat angular motion, providing an accessible platform for exploring the principles of rotational dynamics. The features incorporated into the tool ensure computational efficiency while maintaining pedagogical value, making the tool ideal for both educational settings and preliminary CubeSat design evaluations.

Thus, this study contributes to the growing body of educational tools designed to make knowledge about space technology more accessible and comprehensible for the next generation of space professionals. The presented approach not only serves as an educational resource but also as a means for CubeSat developers to better assess attitude control strategies in early design and mission planning stages.

REFERENCES

[1] A. Toorian, K. Diaz, and S. Lee, “The CubeSat Approach to Space Access,” 2008 IEEE Aerospace Conference. IEEE, Mar. 2008.

[2] A. Poghosyan and A. Golkar, “CubeSat evolution: Analyzing CubeSat capabilities for conducting science missions,” Progress in Aerospace Sciences, vol. 88. Elsevier BV, pp. 59–83, Jan. 2017.

[3] D. Selva and D. Krejci, “A survey and assessment of the capabilities of CubeSats for Earth observation,” Acta Astronautica, vol. 74. Elsevier BV, pp. 50–68, May 2012.

[4] N. Saeed, A. Elzanaty, H. Almorad, H. Dahrouj, T. Y. Al-Naffouri, and M.-S. Alouini, “CubeSat Communications: Recent Advances and Future Challenges,” IEEE Communications Surveys & Tutorials, vol. 22, no. 3. Institute of Electrical and Electronics Engineers (IEEE), pp. 1839–1862, 2020.

[5] G. Benedetti et al., “Interplanetary CubeSats for asteroid exploration: Mission analysis and design,” Acta Astronautica, vol. 154. Elsevier BV, pp. 238–255, Jan. 2019.

[6] G. P. Candini, F. Piergentili, and F. Santoni, “Miniaturized attitude control system for nanosatellites,” Acta Astronautica, vol. 81, no. 1. Elsevier BV, pp. 325–334, Dec. 2012.

[7] M. Ovchinnikov, V. Pen’ko, O. Norberg, and S. Barabash, “Attitude control system for the first swedish nanosatellite ‘MUNIN,’” Acta Astronautica, vol. 46, no. 2–6. Elsevier BV, pp. 319–326, Jan. 2000.

[8] I. V. Belokonov, I. A. Timbai, and P. N. Nikolaev, “Analysis and Synthesis of Motion of Aerodynamically Stabilized Nanosatellites of the CubeSat Design,” Gyroscopy and Navigation, vol. 9, no. 4. Pleiades Publishing Ltd, pp. 287–300, Oct. 2018.

[9] N. Botha and S. Marais, “Development of a Graphical User Interface as a Learning Tool for Artificial Intelligence,” 2021 Rapid Product Development Association of South Africa - Robotics and Mechatronics - Pattern Recognition Association of South Africa (RAPDASA-RobMech-PRASA). IEEE, pp. 1–6, Nov. 03, 2021.

[10] M. Mohd Ali and N. Hashim, “Development of deep learning based user-friendly interface for fruit quality detection,” Journal of Food Engineering, vol. 380. Elsevier BV, p. 112165, Nov. 2024.

[11] F. A. Chiappini, H. C. Goicoechea, and A. C. Olivieri, “MVC1_GUI: A MATLAB graphical user interface for first-order multivariate calibration. An upgrade including artificial neural networks modelling,” Chemometrics and Intelligent Laboratory Systems, vol. 206. Elsevier BV, p. 104162, Nov. 2020.

[12] M. Gasparic, A. Janes, F. Ricci, G. C. Murphy, and T. Gurbanov, “A graphical user interface for presenting integrated development environment command recommendations: Design, evaluation, and implementation,” Information and Software Technology, vol. 92. Elsevier BV, pp. 236–255, Dec. 2017.

[13] P. Mishra et al., “MBA-GUI: A chemometric graphical user interface for multi-block data visualisation, regression, classification, variable selection and automated pre-processing,” Chemometrics and Intelligent Laboratory Systems, vol. 205. Elsevier BV, p. 104139, Oct. 2020.

[14] M. Victoria, O. M. Querin, C. Díaz, and P. Martí, “liteITD a MATLAB Graphical User Interface (GUI) program for topology design of continuum structures,” Advances in Engineering Software, vol. 100. Elsevier BV, pp. 126–147, Oct. 2016.

[15] Y. Errazzouki, A. Habib, A. Jabiri, M. Sabil, and Z. Benkhaldoun, “Developing MATLAB graphical user interface for acquiring single star SCIDAR data,” Astronomy and Computing, vol. 49. Elsevier BV, p. 100878, Oct. 2024.

[16] R. F. Hadi and Dr. Rianto Adhy Sasongko, “CubeSat Dynamic Design and Visualization Tools in MatLab ® Environment,” Unpublished, 2014.

[17] D. S. Ivanov et al., “Software Package for Simulating the Angular and Orbital Motion of a Satellite,” Mathematical Models and Computer Simulations, vol. 12, no. 4. Pleiades Publishing Ltd, pp. 561–568, Jul. 2020.

[18] K. A. Ezzat, L. N. Mahdy, A. E. Hassanien, and A. Darwish, “Robust Simulation and Visualization of Satellite Orbit Tracking System,” Advances in Intelligent Systems and Computing. Springer International Publishing, pp. 505–514, Aug. 29, 2018.

[19] Andrew J. Turner, “The Development and Use of Open-source Spacecraft Simulation and Control Software for Education and Research,” 2nd IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT’06). IEEE, pp. 330–336, July 2006.

[20] M. G. Shirobokov and S. P. Trofimov, “KIAM Astrodynamics Toolbox for Spacecraft Orbital Motion Design,” Programming and Computer Software, vol. 50, no. 1. Pleiades Publishing Ltd, pp. 42–52, Feb. 2024.

[21] I. V. Belokonov, I. A. Timbai, and E. V. Barinova, “Design Parameters Selection for CubeSat Nanosatellite with a Passive Stabilization System,” Gyroscopy and Navigation, vol. 11, no. 2. Pleiades Publishing Ltd, pp. 149–161, Apr. 2020.

[22] Hicks, K. D. Introduction to AstrodynamicReentry. Books Express Publishing, 2009.

[23] E. V. Barinova and I. A. Timbai, “Determining of Equilibrium Positions of CubeSat Nanosatellite under the Influence of Aerodynamic and Gravitational Moments,” 2020 27th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS). IEEE, pp. 1–4, May 2020.