IJSES
Volume 10, Issue 9, pp. 41-46, September 2026.
International Journal of Scientific Engineering and Science
ISSN (Online): 2456-7361

PlanetSim.jl: An Interactive Julia-Based N-Body Simulation Tool for Educational Applications

Kuo-pao Yang[1], Elijah Phifer[2], Erick Diaz[3], Seth Landry[4]
[1, 2, 3, 4]Computer Science Department, Southeastern Louisiana University, Hammond, Louisiana, USA

Abstract*—PlanetSim.jl, an open-source Julia-based package designed to make N-body simulations more accessible for education. N-body simulations are essential tools for studying complex gravitational interactions in astrophysics and physics, yet they are often confined to advanced research due to their computational demands and the lack of intuitive, user-friendly interfaces. PlanetSim.jl addresses this gap by providing an interactive environment where users can define celestial bodies, set initial conditions, run simulations, and dynamically visualize results. By encouraging hands-on experimentation, the platform supports an interdisciplinary learning experience that integrates physics, mathematics, and computational science. PlanetSim.jl emphasizes intuitive exploration of orbital mechanics, numerical methods, and dynamical systems, offering students and educators a practical framework for developing critical thinking and problem-solving skills.

***Keywords*—** *N-body simulations, Julia programming, Educational tools, Virtual labs, Computational modeling.*

I. INTRODUCTION

Technology has increasingly become a powerful tool for expanding access to education. Educators and learners are no longer limited to costly physical manipulatives or time-consuming hands-on models. Among the most effective educational technologies are virtual laboratories and interactive simulators, which allow users to explore and manipulate simulated environments in an engaging and flexible way. Virtual labs provide learners with opportunities to experiment, test ideas, and interact directly with concepts, making abstract material more tangible and visually accessible, which is an especially important benefit for visual learners. By enabling open-ended exploration in a controlled setting, these environments encourage scientific inquiry and active engagement with scientific reasoning.

The teaching of complex physical phenomena, such as gravitational dynamics, is particularly well suited to simulation-based learning. Interactive systems have been shown to enhance conceptual understanding and promote scientific exploration by transforming abstract principles into concrete, visual experiences [1]. However, most existing N-body simulation tools are designed primarily for research applications and remain inaccessible to beginners due to steep learning curves and computational demands, limiting their effectiveness in educational settings.

PlanetSim.jl is a Julia-based N-body simulation tool designed to enhance educational experiences in physics and computational science. Its development addresses several limitations found in existing simulation tools, offering distinct advantages that make it a superior choice for both students and educators.

First, PlanetSim.jl takes advantage of the high-performance capabilities of Julia, a programming language known for combining the execution speed of low-level languages such as C with the expressive simplicity of high-level languages like Python. This balance allows users to develop efficient and readable code without compromising usability. In an educational context, students can interact with complex simulations without being burdened by complicated syntax or performance limitations. Educators similarly benefit from Julia's clear syntax and powerful features, which make it easier to design, adapt, and extend simulations that demonstrate sophisticated physical concepts. Moreover, Julia's growing adoption in scientific computing ensures access to a strong community and a wide range of learning resources.

Second, PlanetSim.jl incorporates interactive features that are essential for effective learning. Interactive simulations are widely recognized as powerful educational tools because they actively engage students and allow real-time manipulation of variables, which supports deeper understanding and helps address common misconceptions [2][3]. Through real-time visualization, users can immediately observe the effects of changes in initial conditions, promoting a stronger intuition for dynamic systems. This level of interactivity encourages exploration and hypothesis testing, as students can experiment with different scenarios and directly observe the resulting behavior. Such hands-on engagement is particularly valuable for understanding complex concepts such as gravitational interactions and orbital mechanics. By offering an immersive environment for experimentation, PlanetSim.jl translates abstract theoretical principles into concrete, accessible learning experiences.

Third, PlanetSim.jl features a modular design that enables users to customize and extend simulations to meet specific educational objectives. This flexibility is especially valuable for educators who wish to align simulations with particular curricula, as well as for students interested in exploring targeted aspects of N-body dynamics. As an open-source platform, PlanetSim.jl encourages users to engage directly with its codebase, fostering transparency and collaborative development. This openness helps bridge the gap between theoretical concepts and practical implementation while supporting the development of computational skills. By interacting with the underlying code, users gain deeper insight into simulation techniques and numerical methods, strengthening their understanding of computational modeling.

PlanetSim.jl stands out as a superior educational tool by combining Julia's high performance and user-friendly syntax

with interactive features and a customizable, open-source framework. These features make the tool approachable and useful for a diverse audience that effectively unifies theoretical concepts and practical application in the study of N-body simulations.

II. Related Work

The N-body problem is a fundamental challenge in physics and astronomy, involving the prediction of individual motions of a group of celestial objects interacting through mutual gravitational forces. While the two-body problem has exact solutions, extending this to three or more bodies introduces complexities that often render analytical solutions infeasible. This complexity arises from the nonlinear nature of gravitational interactions, leading to chaotic behavior in systems with multiple bodies. Understanding such dynamics is crucial for studying planetary systems, star clusters, and galactic formations.

To address the complexities of the N-body problem, computational tools have become indispensable. These tools employ numerical methods to approximate the solutions of the equations governing gravitational interactions. By discretizing time and calculating forces and subsequent motions iteratively, simulations can model the evolution of complex systems over time. This computational approach allows researchers to explore scenarios that are analytically intractable, providing insights into the formation and evolution of various astronomical structures.

Several educational tools have been developed to make the concepts of gravitational dynamics more accessible to students and educators. For instance, studies demonstrate that simulation-based platforms, such as PhET and other interactive environments, connect theoretical concepts with practical implementation between theoretical concepts and practical application, enabling learners to engage actively with dynamic systems [4][5]. The PhET Interactive Simulations project offers a "Gravity and Orbits" simulation, enabling users to visualize and manipulate planetary motions. While user-friendly, such tools often oversimplify the user's interaction, limiting the depth of exploration into complex N-body interactions.

Another example is the "My Solar System" simulation provided by The PhET Interactive Simulations project, which allows users to adjust parameters and observe resulting orbital behaviors. However, this tool primarily focuses on two-dimensional simulations and lacks the functionality for the user to be able to interact with the simulation in three dimensions.

In the realm of more advanced simulations, software like REBOUND offers high-precision N-body simulations used in professional research [6]. While powerful, such tools often require substantial computational resources and a steep learning curve, making them less accessible for educational purposes.

PlanetSim.jl distinguishes itself by addressing these limitations. Developed in Julia, it combines high performance with an accessible syntax, making it suitable for both educational and research applications. Its interactive features, including real-time visualization and the ability to experiment with initial conditions, encourage exploration and hypothesis testing. The modular and open-source structure of PlanetSim.jl enables users to customize simulations and delve into computational techniques, effectively bridging the gap between theoretical concepts and practical applications.

III. Implementation

A. *Initialization of the Simulation*

The simulation process in PlanetSim.jl begins with the initialization of the environment using the initialize simulation function. This function requires two inputs: the gravitational constant and the time step. The gravitational constant serves as a scaling factor for the forces calculated between celestial bodies, while the time step defines the intervals at which the simulation updates the positions and velocities of the planets. Smaller time steps provide more precise and detailed simulations, particularly for systems with rapid dynamics or close encounters, but they come at the cost of increased computational load. Larger time steps, on the other hand, allow for faster simulations but may introduce errors, especially in scenarios requiring high accuracy. This tunable parameter provides flexibility for users to balance precision and performance, catering to different educational and research needs.

B. *Defining and Managing Celestial Bodies*

Celestial bodies, referred to as "planets," are the fundamental units of the simulation. Each planet is defined using attributes such as its name, mass, position, and velocity. These attributes allow the simulator to calculate gravitational interactions and update the motion of each body over time. Planets are encapsulated as modular Planet objects, making them easy to manage and integrate into the simulation. Users can add planets to the simulation using the add_planet! function or remove them with the remove_planet! function. This feature enables users to conduct experiments, such as analyzing the effect of removing a key celestial body like Jupiter to study the resulting instability in the solar system. Such scenarios are particularly useful for understanding and visualizing the delicate balance of gravitational forces.

C. *Gravitational* Force Calculations

The core of PlanetSim.jl lies in its implementation of algorithms to calculate gravitational forces and update the motion of celestial bodies. The simulator supports two algorithms: the direct pairwise method and the Barnes-Hut algorithm, each suited to different scales and computational needs.

1) *Direct Pairwise Calculation*

The direct pairwise method computes gravitational forces by iterating through all pairs of celestial bodies and computing the force using Newtonian mechanics. This method provides high accuracy but becomes computationally expensive as the number of bodies increases, scaling with $O(N^2)$ complexity. For smaller systems, such as the solar system, the direct pairwise approach is ideal, offering precise results within a reasonable timeframe. While this method is ideal for smaller

International Journal of Scientific Engineering and Science
Volume 10, Issue 9, pp. 41-46, September 2026. ISSN (Online): 2456-7361

systems like the solar system, it becomes infeasible for larger systems due to its high computational cost.

This direct pairwise method calculates gravitational forces by iterating over all pairs of celestial bodies. As illustrated in Fig. 1, the force on each body is determined using the Hamiltonian for the N-body problem [7]:

$$H(p,q) = \sum_{i=0}^{N} \frac{p_i^2}{2m_i} - G \sum_{i=1}^{N} \sum_{j=0}^{i-1} \frac{m_i m_j}{\|q_i - q_j\|}$$

Fig. 1. The N-Body Problem's Hamiltonian

In the context of the N-body problem, the Hamiltonian function H encapsulates the total energy of a system comprising N interacting bodies. It is expressed as the sum of the kinetic and potential energies of the system.

2) Barnes-Hut Algorithm

To address the limitations of the direct pairwise method, PlanetSim.jl incorporates the Barnes-Hut algorithm, which significantly improves computational efficiency for large-scale simulations. The Barnes-Hut algorithm shown in Fig. 2 reduces complexity to O(NlogN) by approximating the gravitational effects of distant bodies using hierarchical clustering [8]. The algorithm operates by constructing a tree structure, often called an octree in 3D simulations. Each node in the tree represents a region of space, and the bodies within each region are recursively subdivided into smaller regions until each region contains at most one body.

```
function Barnes_Hut(simulation):
    # Step 1: Build the quadtree
  root = buildQuadtree(simulation.bodies, simulation.boundary)

    # Step 2: Calculate mass properties for the tree
     calculateMassProperties(root)

    # Step 3: Compute forces on each body
     for body in bodies:
       computeForces(simulation.body, root, simulation.theta)
```

Fig. 2. Barnes-Hut Algorithm

At each level of the tree, clusters of bodies are represented by a single "center of mass" and a combined total mass. When calculating the force on a body, the algorithm decides whether to approximate the force from a distant cluster using the cluster's center of mass or to further subdivide the cluster into smaller groups. This decision is governed by a threshold parameter θ, which balances speed and accuracy. A smaller θ results in higher accuracy at the cost of additional calculations, while a larger θ speeds up the simulation by accepting coarser approximations. This trade-off makes the Barnes-Hut algorithm particularly effective for simulations involving thousands or even millions of bodies, such as modeling star clusters or galaxies.

D. Visualization Capabilities

Visualization is a cornerstone of PlanetSim.jl, providing users with powerful tools to interpret and analyze simulation data. Visualization-based interactive systems have been shown to significantly improve learning outcomes by presenting abstract phenomena in intuitive and engaging ways, making them more accessible to learners at various levels [9]. The simulator offers several visualization options tailored to different needs:

1) Static 3D Plots

These plots provide snapshots of planetary orbits, helping users analyze stable configurations or identify anomalies in motion. They are particularly useful for presenting results in educational or research contexts shown in Fig. 3.

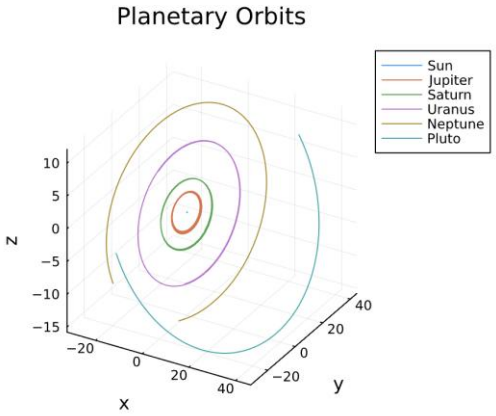

Fig. 3. Static 3D plot of planets' orbit paths

2) Animated GIF Exports

For long-term studies or presentations, simulations can be exported as animated GIFs, offering a convenient way to share and review results shown in Fig. 4.

3) Interactive 3D Animations

Users can explore planetary orbits dynamically, navigating through the simulation space to observe the evolution of the system over time. This feature is valuable for allowing the user to be engaged with the simulation in a more hands-on manner shown in Fig. 5.

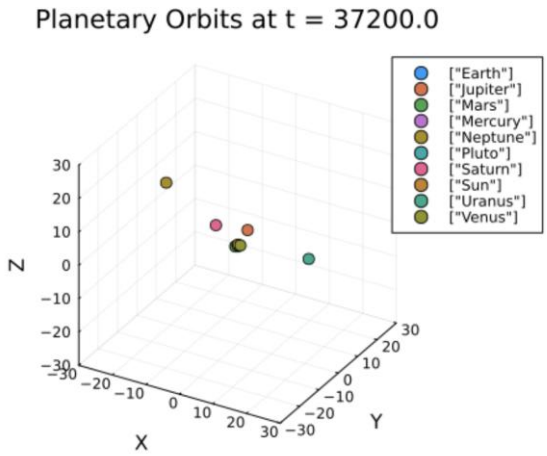

Fig. 4. Screenshot of 3D GIF animation of planets orbits around the sun

International Journal of Scientific Engineering and Science
Volume 10, Issue 9, pp. 41-46, September 2026. ISSN (Online): 2456-7361

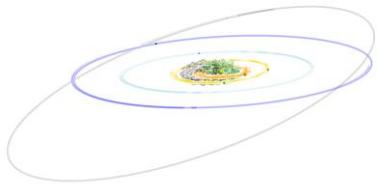

Fig. 5. Interactive 3D animation of planet orbits that allow a user to tilt and rotate to view the simulation from different perspectives

4) Interactive 3D Playground with Mass Adjustment Tool

A unique visualization feature allows users to manipulate the Sun's mass dynamically using a slider. This tool lets users observe and analyze the impact of mass changes on the solar system's stability in real time, providing a prepackaged experiment to gain a hands-on learning experience shown in Fig. 6.

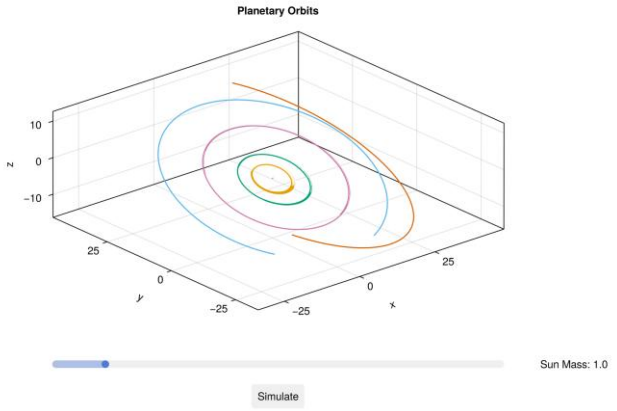

Fig. 6. Interactive 3D animation of planet orbits that allow a user to tilt and rotate to view the simulation from different perspectives and change the mass of the sun within the simulation

E. Design Philosophy and Extensibility

The design of PlanetSim.jl emphasizes modularity and openness, making it a versatile tool for both educational and research applications. The transparent codebase encourages users to explore the implementation, modify existing features, or add new functionalities. For example, advanced users can integrate additional numerical methods, implement custom visualization techniques, or incorporate real-world astronomical datasets into their simulations. This flexibility ensures that PlanetSim.jl remains adaptable to evolving educational and research needs.

By combining robust algorithms, interactive visualizations, and user-friendly design, PlanetSim.jl provides a comprehensive framework for exploring celestial dynamics. Its dual-algorithm approach ensures that the tool caters to both small-scale and large-scale simulations, while its modular architecture invites innovation and customization. The implementation of PlanetSim.jl exemplifies how computational tools can close the gap between theoretical learning and practical application, empowering users to delve deeper into the intricacies of gravitational systems.

IV. Evaluation

The evaluation of PlanetSim.jl demonstrates its effectiveness as both an educational tool and a powerful simulator for studying planetary dynamics. To thoroughly assess its capabilities, the tool was benchmarked by comparing the average runtimes between algorithms and between Python and Julia.

PlanetSim.jl was evaluated in two primary contexts: computational efficiency between algorithms and runtime between both Python and Julia implementations of the algorithms. These tests provided insights into its runtime, accuracy, and scalability using its three core algorithms: the direct pairwise method (with and without the Hamiltonian) and the Barnes-Hut algorithm. Fig. 7 shows the comparison in runtime between the three algorithms and compares the runtimes between Python and Julia. As seen, Julia consistently outperforms Python. Runtime was compared across algorithms by running a simulation with 2 bodies and the same starting conditions with each algorithm.

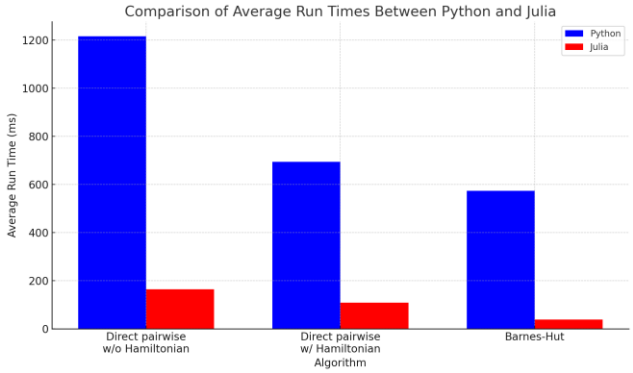

Fig. 7. Benchmarks for all algorithms implemented and the comparison of run time in Python (blue) versus Julia (red)

PlanetSim.jl's visualization features further set it apart. Static 3D plots provided clear snapshots of planetary orbits, enabling users to analyze stable configurations. Interactive 3D animations allowed real-time exploration of orbital dynamics, offering a hands-on way to understand complex phenomena. One standout feature is the mass slider tool, which lets users dynamically adjust the Sun's mass and immediately observe the effects on the solar system's stability. Such interactive and hands-on tools are aligned with findings that real-time simulation environments encourage exploration and allow users to visualize and comprehend complex dynamics effectively [10][11]. This level of interactivity not only engages users but also fosters a deeper understanding of the delicate balance within gravitational systems.

Compared to other solutions, PlanetSim.jl strikes a unique balance between accessibility and functionality. Tools like PhET Interactive Simulations [12] are user-friendly but limited to two-dimensional representations, while high-precision platforms like REBOUND cater to advanced researchers but require significant computational expertise.

PlanetSim.jl solves this by combining an intuitive interface with robust computational power. Its dual-algorithm approach allows users to tailor simulations to their needs, offering precision for small systems and efficiency for large-scale models. Additionally, its modular and open-source design encourages users to adapt the tool for their own educational or research purposes, ensuring that it remains versatile and forward-looking.

What makes PlanetSim.jl particularly impressive is its ability to seamlessly blend performance with usability. The intuitive interface and clear documentation make it accessible to students and educators, while its computational capabilities meet the demands of researchers studying large-scale systems. Visualization tools enhance their educational value, transforming abstract theoretical concepts into tangible learning experiences.

Overall, PlanetSim.jl outperforms many existing tools by providing a comprehensive platform that caters to a wide range of users. Its efficient algorithms, engaging visualizations, and adaptability ensure that it not only meets but exceeds expectations for a simulation tool in this domain. The inclusion of visualizations and interactive features reinforces its position as a leading solution for exploring celestial mechanics. This combination of accuracy, efficiency, and usability makes PlanetSim.jl a remarkable achievement, redefining how simulations can be used to understand and teach gravitational dynamics.

V. Conclusion and Future Work

PlanetSim.jl represents a significant advancement in the field of educational and computational tools for simulating planetary systems. It provides an innovative platform that helps the user create a link between theoretical concepts and practical application, offering users the ability to simulate and visualize complex gravitational dynamics interactively. The combination of Julia's high-performance capabilities and its user-friendly syntax ensures that PlanetSim.jl is both accessible to beginners and powerful enough for advanced research. This makes it a valuable tool for a wide audience, including students looking to grasp the fundamentals of celestial mechanics, educators seeking to create engaging classroom experiences, and researchers requiring a flexible framework for modeling intricate gravitational systems.

The modular design of PlanetSim.jl, paired with its dual algorithm support and robust visualization features, makes it adaptable to a variety of use cases. The direct pairwise algorithm offers precise calculations for smaller systems, such as the solar system, while the Barnes-Hut algorithm allows efficient simulation of larger systems. These capabilities, combined with interactive visualization tools - including static 3D plots, animated visualizations, and the innovative mass slider for dynamic experimentation - equip users to explore the dynamic behavior of planetary systems in both intuitive and rigorous ways. By transforming abstract equations and models into tangible, interactive experiences, PlanetSim.jl fosters deeper understanding and engagement.

As an open-source framework, PlanetSim.jl goes beyond being just a simulation tool; it serves as a collaborative platform that invites users to contribute to its growth and evolution. The transparency of its codebase and the extensibility of its design empower educators to customize simulations for specific lessons, students to experiment with new ideas, and researchers to extend its functionality to meet the needs of specialized studies. This collaborative ethos ensures that PlanetSim.jl will continue to evolve as a resource for learning and discovery.

The implementation of PlanetSim.jl reflects a thoughtful integration of computational precision and accessibility. By leveraging interactive features and intuitive design, the tool embodies principles highlighted in studies on virtual labs and educational technologies, which emphasize the role of interactivity in promoting deeper understanding and engagement. Its ability to handle complex gravitational systems while maintaining an interactive and user-friendly interface makes it a standout resource for exploring celestial mechanics. The tool bridges the gap between theoretical understanding and practical application, enabling users to test hypotheses, analyze dynamic behaviors, and develop an appreciation for the complexities of planetary motion. This synthesis of functionality and usability ensures that PlanetSim.jl is not just a tool for computation but also a medium for fostering curiosity and critical thinking.

Looking to the future, PlanetSim.jl is poised to expand its capabilities further. Upcoming developments aim to improve simulation accuracy through adaptive time-stepping methods, which are particularly useful for scenarios involving highly dynamic or near-collision events. Expanding the suite of algorithms to include options such as symplectic integrators will provide users with even more tools to explore diverse gravitational phenomena. On the visualization front, incorporating augmented reality (AR) and virtual reality (VR) technologies has the potential to revolutionize how users interact with simulations, offering immersive experiences that make complex dynamics more comprehensible. Additionally, integrating real-world astronomical datasets will enable users to simulate and analyze observed celestial events, adding a layer of real-world applicability to the platform.

These planned enhancements will solidify PlanetSim.jl's position as a versatile and engaging platform for exploring the intricacies of gravitational systems. By empowering users to move seamlessly from theoretical models to hands-on experimentation, PlanetSim.jl not only enhances the understanding of celestial mechanics but also inspires creativity and collaboration. Its blend of computational rigor, educational value, and future-oriented design ensures that PlanetSim.jl will remain a cornerstone tool for both teaching and research in the years to come.

References

[1] Z. Guo, "Interactive Teaching Model Reform Based on Network Platform and Virtual Simulation Technology," *2022 International Conference on Education, Network and Information Technology (ICENIT)*, Liverpool, United Kingdom, pp. 17-23, DOI: 10.1109/ICENIT57306.2022.00012, September 2022.

[2] R. SenthilKumar, "Work in Progress: Use of Interactive Simulations in the Active Learning Model in Physics Education for Engineering Students at a College in Oman," *2019 IEEE Global Engineering*

Education Conference (EDUCON), Dubai, United Arab Emirates, pp. 1359-1362, DOI: 10.1109/EDUCON.2019.8725215, April 2019.

[3] O. Ameerbakhsh, S. Maharaj, A. Hussain, T. Paine, and S. Taiksi, "An exploratory case study of interactive simulation for teaching Ecology," *2016 15th International Conference on Information Technology Based Higher Education and Training (ITHET)*, Istanbul, Turkey, pp. 1-7, DOI: 10.1109/ITHET.2016.7760725, September 2016.

[4] B. K. Samanthula, M. Mehran, M. Zhu, N. Panorkou, and P. Lal, "Experiences Toward An Interactive Cloud-Based Learning System for STEM Education," *2020 IEEE Integrated STEM Education Conference (ISEC)*, Princeton, NJ, USA, pp. 1-6, DOI: 10.1109/ISEC49744.2020.9280688, August 2020.

[5] K. Jankiewicz, P. Migdal, and P. Grabarz. "Virtual Lab by Quantum Flytrap: Interactive Simulation of Quantum Mechanics," *CHI Conference on Human Factors in Computing Systems Extended Abstracts (CHI EA '22)*, New Orleans, LA, USA, Article No. 175, pp. 1 – 4, DOI: 10.1145/3491101.3519885, February 2022.

[6] Rebound, https://rebound.hanno-rein.de, December 2025.

[7] C. Rackauckas and Q. Nie, "DifferentialEquations.jl: A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia," *Journal of Open Research Software*, Vol. 5, Issue 1, Article 15, pp. 1-10, DOI: 10.5334/jors.151, May 2017.

[8] J. Barnes and P. Hut, "A Hierarchical O(N log N) Force-Calculation Algorithm," *Nature Publishing Group*, Vol. 324, pp. 446–449, DOI: 10.1038/324446a0, December 1986.

[9] Z. Huanyin, L. Jinsheng, W. Yangjie, X. Hong, and Q. Min, "Computer Simulation for Undergraduate Engineering Education," *2009 4th International Conference on Computer Science & Education, Nanning*, pp. 1353-1356, DOI: 10.1109/ICCSE.2009.5228177, August 2009.

[10] F. V. Kowalski and S. E. Kowalski, "The Effect of Student Learning Styles on the Learning Gains Achieved When Interactive Simulations Are Coupled with Real-Time Formative Assessment Via Pen-Enabled Mobile Technology," *2012 Frontiers in Education Conference Proceedings*, Seattle, WA, USA, pp. 1-5, DOI: 10.1109/FIE.2012.6462281, October 2012.

[11] H. Zhou, "Interactive Simulation Teaching System of Electrical Engineering Based on Virtual Reality," *2021 2nd International Conference on Artificial Intelligence and Education (ICAIE)*, Dali, China, pp. 119-122, DOI: 10.1109/ICAIE53562.2021.00033, June 2021.

[12] K. Perkins, W. Adams, M. Dubson, N. Finkelstein, S. Reid, C. Wieman, and R. LeMaster. "PhET: Interactive Simulations for Teaching and Learning Physics," *The Physics Teacher*, Vol. 44, Issue 1, pp. 18–23, DOI: 10.1119/1.2150754, 2006.