universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Solar System Game In Education

verfasst von | submitted by

Dongzhou Fang

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2025 | Vienna, 2025

Studienkennzahl lt. Studienblatt | degree programme code as it appears on the student record sheet:	UA 066 921
Studienrichtung lt. Studienblatt | degree programme as it appears on the student record sheet:	Masterstudium Informatik | Computer science
Betreut von | Supervisor:	Univ.-Prof. Dipl.-Ing. Dr. Helmut Hlavacs

Acknowledgements

I would like to express my deepest gratitude to my family and friends for their support and encouragement throughout the development of my video game and the writing of this thesis. Their presence and belief in me have been a constant source of strength. I am also thankful to my supervisor, Prof. Dr. Helmut Hlavacs, for his continuous guidance and insightful feedback, and he always gives me valuable advice. His willingness to listen and help refine my ideas has been instrumental in shaping the outcomes of this work. Finally, I would like to thank every member of my social circle who tests, gives thoughtful feedback, and shows interest in this game, they have served as invaluable pillars of support throughout this development.

Abstract

In recent years, game-based learning (GBL) has emerged as a powerful tool for enhancing education by integrating interactivity, engagement, and knowledge acquisition. This thesis presents the development of an educational game focused on the solar system, combining scientific accuracy with immersive gameplay. The project aims to create an interactive learning environment where players explore planetary science while defending the solar system's resources through strategic missions.

This research examines key aspects of educational game design, including game mechanics, virtual environments, and scientific simulations. By leveraging real-world astronomical data, the game provides an accurate representation of celestial bodies, planetary atmospheres, and orbital mechanics. Additionally, the integration of narrative-driven missions ensures that learning is embedded within engaging gameplay rather than being separate from it.

The study also addresses challenges in balancing scientific accuracy with user engagement, optimizing the game's interface for effective information delivery, and evaluating its educational impact through user testing. The results demonstrate the potential of serious games to enhance STEM education by making complex scientific concepts more accessible and enjoyable. This research contributes to the growing field of digital education, showcasing how interactive technologies can transform learning experiences in astronomy and beyond.

The development of this game aims to address the following research questions: (i) How to create a realistic game environment? (ii) How can astronomical knowledge be effectively conveyed within a game environment? (iii) How can a balance be struck between scientific accuracy and game playability? (iv) Can this game enhance players' understanding and interest in solar system knowledge? (v) How can the impact of the game on learning outcomes be assessed?

To address the research questions, the game was developed based on the real astronomical datasets, user-centered design. A combination of realistic simulations, tasks with a storyline, and exploratory gameplay allowed the integration of educational content into engaging game mechanics. Through performance testing and a user questionnaire, we evaluated the game's impact on users' knowledge acquisition and interest in planetary science. The evaluation indicates that the game significantly enhances learners' conceptual understanding of spatial relationships and planetary characteristics about the solar system, while also increasing engagement and motivation. The balance between scientific accuracy and gameplay was achieved through simplified but realistic simulations and carefully designed missions. Overall, the game demonstrates the effectiveness of game-based learning in delivering complex scientific knowledge and offers valuable insights for future educational game development in STEM education.

Kurzfassung

In den letzten Jahren hat sich Game-Based Learning (GBL) als leistungsstarkes Instrument zur Verbesserung von Lernprozessen erwiesen, indem es Interaktivität, Engagement und Wissenserwerb miteinander verbindet. Die vorliegende Arbeit präsentiert die Entwicklung eines edukativen Spiels über unser Sonnensystem, das wissenschaftliche Genauigkeit mit spannendem Gameplay kombiniert. Ziel des Projekts ist es, eine interaktive Lernumgebung zu schaffen, in der die Spielenden viele Fakten über die Planeten unseres Sonnensystems lernen und dabei in einem Actionspiel in strategischen Missionen die Ressourcen des Sonnensystems verteidigen.

Die hier präsentierte Forschung untersucht die zentralen Aspekte des Designs des Spiels, einschließlich Spielmechaniken, virtueller Umgebungen und wissenschaftlicher Simulationen. Durch die Nutzung realer astronomischer Daten bietet das Spiel eine genaue Darstellung von Himmelskörpern, Planetenatmosphären und Orbitalmechanik. Darüber hinaus sorgt die Integration erzählgetriebener Missionen dafür, dass das Lernen in das unterhaltsame Spielerlebnis eingebettet ist, anstatt davon getrennt zu sein.

Die Entwicklung des Spiels zielt darauf ab, die folgenden Forschungsfragen zu beantworten: (i) Wie kann eine realistische Spielumgebung geschaffen werden? (ii) Wie lässt sich astronomisches Wissen effektiv in ein Spiel integrieren? (iii) Wie kann ein Gleichgewicht zwischen wissenschaftlicher Genauigkeit und Spielbarkeit erreicht werden? (iv) Kann dieses Spiel das Verständnis und Interesse der Spielenden am Wissen über das Sonnensystem verbessern? (v) Wie kann die Wirkung des Spiels auf die Lernergebnisse gemessen werden?

Zur Beantwortung dieser Forschungsfragen wurde das Spiel auf Grundlage realer astronomischer Datensätze und benutzerzentrierten Designs entwickelt. Eine Kombination aus realistischen Simulationen, aufgabenbasiertem Storytelling und explorativem Gameplay ermöglichte die Integration von Bildungsinhalten in motivierende Spielmechaniken. Durch Leistungstests und eine Nutzerbefragung wurde die Wirkung des Spiels auf den Wissenszuwachs und das Interesse an Planetenwissenschaft untersucht. Die Evaluation zeigt, dass das Spiel das konzeptionelle Verständnis der Lernenden für räumliche Beziehungen und planetare Eigenschaften des Sonnensystems signifikant verbessert und gleichzeitig Motivation und Engagement steigert.

Insgesamt demonstriert das Spiel die Wirksamkeit von Game-Based Learning bei der Vermittlung komplexer wissenschaftlicher Inhalte und liefert wertvolle Erkenntnisse für die zukünftige Entwicklung von Lernspielen im MINT-Bereich.

Contents

List of Tables

List of Figures

1 Introduction

Humans are always curious about exploring the universe and understanding its place within it. As the extension of our home in the universe, the solar system not only inspires human imagination but also plays a foundational role in fostering scientific cognition and advancing the fields of STEM (Science, Technology, Engineering, Mathematics). Gaining knowledge about the solar system is a crucial step to understanding the broader universe.

However, the study of the solar system also faces some challenges. One of the primary obstacles is the abstract nature of its concepts, particularly when dealing with scale and distance. People often struggle to imagine the vast spatial distance between planets and the immense sizes of celestial bodies. There are no appropriate media to help people transition from abstract concepts of the solar system to concrete spatial visualization of the universe. Without effective conceptual bridges, it is hard to understand the immense scale and complexity of concepts about the universe [1]. Besides scale, the dynamic and interconnected nature of the solar system such as celestial bodies' motion under the influence of unseen forces, this may bring people more problems for understanding.

Moreover, core complex astrophysical concepts such as gravity and planetary motion like orbits and rotation, are particularly difficult for students without direct and observable experience. Unlike phenomena on Earth that can often be demonstrated or directly interacted with, planetary movements occur over vast timescales and distances, making direct observation by the naked eye is a huge challenge. Consequently, constructing astronomical knowledge becomes an abstract and challenging task.

Astronomy, as a discipline, it is almost entirely based on observation and inference. However, except for some celestial bodies such as the Moon, visible planets like the Earth, and observable stars in the night sky, the universe remains imperceptible to human senses [1]. Also, the ability to infer three-dimensional spatial relationships from one or two-dimensional symbolic resources like diagrams or animations is essential for learning astronomy, but this skill is difficult for students to acquire effectively [2]. These limitations will be obstacles for forming a clear and accurate imaging model of the solar system, which may result in misunderstandings about planetary distances, celestial bodies' sizes, and orbital paths.

Addressing these challenges requires innovative educational approaches that can bridge the gap between abstract concepts and concrete understanding. Traditional methods are foundational and easy to teach, but they often focus more on static diagrams and verbal explanations, which may not be sufficient for students to develop the necessary spatial reasoning skills or intuitive understanding of the scale and motion. Researchers and educators notice that we need such tools can make these abstract concepts more concrete, interactive, and engaging, so that they can promote deeper learning and reduce some common misunderstandings. Recent research in STEM education highlights the

effectiveness of active learning strategies, personalized instruction, and the use of digital tools and simulations to enhance student engagement and understanding of complex concepts, cause these concepts are difficult to observe directly or too abstract [3].

With the development of technology, Game-Based Learning (GBL) has gained widespread attention in the field of education [4][5]. In recent years, digital educational tools have gradually become an effective supplement to traditional classroom teaching [6][7][8][9]. Among these, educational games based on virtual environments have been proven to enhance learners' interest and better for keeping their knowledge. Astronomy is an important topic in basic science education, but due to the abstract nature of the astronomical concepts involved, and hard for experiencing them, many students face difficulties in understanding and observing the concepts [1].

Against this background, this project designs and develops an educational game focused on the solar system. The core objective is to create an engaging and effective platform based on Game-Based Learning, where students can explore, interact with, and learn about the solar system. Through active engagement in the game, students will gradually master knowledge about planets, moons, and other celestial bodies, and the fundamental laws about the solar system's motion, such as Kepler's laws. The focus of this project is to explore how gamification, through carefully designed mechanics and content, it can effectively enhance students' understanding of the complex and fascinating topic of the solar system, specifically targeting the challenges related to abstraction, scale, distance, and the lack of direct observation. By providing a dynamic and interactive virtual environment, the game allows students to observe and explore the aspects of the solar system.

Moreover, many educational games have the difficulty about how to get a balance between scientific rigor and entertainment [10][11]. As a result, the experience is either too boring due to lack of engagement, they more focus on the educational content [12], or too focused on entertaining, which causes inefficient learning. Therefore, this project tries to integrate a task system, storyline, and interactive exploration mode, this will enable players to learn and understand scientific concepts of the solar system through an immersive gaming experience. The design considers integrating the learning objectives within the core gameplay, this would ensure that scientific concepts are not only presented as facts but are integral for development within the game world. This method combines entertainment and education well, it makes games more meaningful and makes learning more interesting.

Game-based learning is fundamentally aligned with constructivist learning theories, which emphasize that learners construct their own understanding and knowledge through experiencing things and reflecting on those experiences [8]. It provides a highly motivating approach, encourages active reflection on actions and outcomes, provides immediate feedback, offers players a sense of control over their own learning path, and allows for process monitoring and replayability.

Within the context of this game, a structured mission system can make players' attention focus on task-relevant information and scientific principles, guiding their learning journey through challenges that build conceptual understanding. A storyline can provide

context and narrative coherence, making the acquisition of knowledge more engaging and reasonable by embedding scientific exploration within a purposeful mission. Interactive exploration is supported by accurate simulations and a virtual environment of the solar system, which allows players to directly interact with celestial bodies, and observe the environment, thereby giving them deeper understanding of abstract concepts like orbital mechanics through direct experience and observation. This dynamic interaction and immediate feedback are particularly valuable for developing intuitive understanding and correcting misconceptions related to motion and forces in space.

By providing a platform with active interaction and visualization, this educational game provides a powerful complement to traditional teaching methods, making the solar system more accessible and understandable to a wider audience.

2 Related Work

This chapter reviews existing research and developments that are directly relevant to the subject matter of this thesis. The main focus is the development of an innovative educational game about the solar system, which integrates engaging space combat mechanics to provide a unique learning experience through immersive virtual environments. To build a strong foundation for this research and to clearly describe the contribution of this thesis, it is essential to examine some key areas. Firstly, explore the current state of research and the potential educational applications within the field of Virtual Environments and Immersive Learning. Secondly, survey related technologies and practices about the scientific simulation of the Solar System, which is crucial for ensuring the scientific accuracy of the game content. Finally, analyze existing solar system educational games to understand their characteristics, strengths, and potential improvement, so that can provide valuable insights for the game design in this project. By organizing and analyzing this related work, this chapter aims to get a better understanding of the current research and highlight the significance of the research.

Virtual environments (VE) provide powerful tools for immersive learning, allowing students to engage with complex concepts in an interactive and intuitive way. These environments are particularly valuable in science education, where they enable learners to visualize and manipulate abstract concepts that are otherwise difficult to grasp through traditional methods [13]. This section explores the role of VE in education, its effects on learning, and notable applications.

Virtual Environments enhance learning by providing immersive simulations that allow learners to visualize and interact with complex concepts. Immersive technologies have great potential to improve learning outcomes and engagement in science education programs. These technologies not only engage students' interests, but also promote deeper understanding and retention of complex material. Virtual environments can facilitate learning in ways that the traditional classroom cannot. For example, by accurately mirroring natural environments, or create unnatural learning environments that can motivate students and enhance the learning experience [13].

In astronomy education, these virtual environments facilitate learning in astronomy education by enhancing understanding in several key areas. They help develop spatial awareness by allowing learners to visualize and understand the relative positions and movements of celestial bodies. Furthermore, VEs help to promote scale perception, enabling users to observe and explore the vast astronomical distances and the immense sizes of celestial bodies. Crucially, these environments support conceptual learning by providing intuitive ways to explore complex physics features, such as orbital mechanics and fundamental astrophysics principles, these may be difficult to study through traditional methods [14].

Some examples of virtual environments utilized in astronomical education include Google Earth VR, which enables users to explore planetary surfaces and celestial landscapes, and the Solar System Scope, it is a 3D model that visually represents planetary orbits and astronomical events. Some other such VR apps that allow users can virtually visit the international space station, explore the surfaces of Mars and the Moon, and even experience the birth of a star. These experiences often utilize spatial audio and haptic feedback to further enhance immersion, making the learning experience more engaging and memorable [15].

Studies have demonstrated that immersive learning environments can significantly enhance cognitive outcomes. By enabling learners to interact actively within a virtual space, these environments allow for a more intuitive grasp of complex concepts. Specifically, immersive experiences facilitate conceptual understanding, as learners can grasp abstract scientific concepts more effectively through direct and interactive exploration [14][16][17]. Furthermore, immersive environments are shown to improve memory retention, as actively experiencing the concepts helps information to be retained more effectively than through passive methods [18]. Immersion can also promote a deeper emotional connection to the subject, leading to increased motivation and a more positive attitude towards learning science. The feeling of "being there" in a virtual environment can create a sense of wonder and curiosity, this may be difficult to achieve through traditional learning methods [14][18][19].

Additionally, studying in virtual environments like VR can enhance learning, and the interaction can influence efficiency and accuracy depending on the task. These environments can significantly enhance concepts understanding in astronomy education by providing immersive learning experiences that make abstract concepts easier to understand. Learners would become active participants, for example, virtually traveling to a star and measuring its properties or experiencing the microgravity of an astronaut. VEs like VR also provide unique opportunities to explore abstract concepts and phenomena beyond the visual limitations of the real world, such as dynamically manipulating the scale, rotation and viewpoints of planets [17][18][19].

Beyond these benefits, immersive environments are also highly effective in increasing learner engagement and motivation. By offering dynamic and interactive experiences, particularly in special methods like VR or AR, these environments can make learning scientific subjects more active and less passive [16]. This increased interest and motivation usually come from the engaging nature of immersive exploring content like a game. The use of game-like elements in virtual learning environments, often known as gamification, it can further enhance engagement by incorporating elements such as challenges, rewards, and competition. This can transform learning from a passive reception of information to an active and enjoyable process [14][18][19].

Despite the potentials, the implementation of virtual environments for education also faces challenges. These include the cost and accessibility of necessary hardware for high-quality experiences, the potential for cognitive overload or motion problems in poorly designed immersive environments, and the complexity and cost of high-quality developing, and educational interactive content. Ensuring that the interactive and immersive elements

directly contribute to specific learning objectives rather than serving only as engaging distractions is also a key challenge, which requires good design and evaluation [12][16][20].

In addition, consideration of math concepts is critical as well. Educators need to be trained to effectively integrate virtual environments into their teaching practice, and course materials need to be designed to take advantage of the capabilities of these technologies. The novelty effect of VR would also decrease with time running, so it is important to design learning experiences that remain engaging over time. Research has shown that traditional teacher education is overly focused on theoretical learning, this may limit opportunities for practical training, which may affect teachers' ability that apply the skills they need in real-world teaching. Therefore, a comprehensive overview of the specific aspects of virtual environment for teacher professional development is important to facilitate teacher professional development [12][20].

Simulating the solar system with high scientific accuracy is very important for educational purposes, as it allows learners to understand complex astronomical phenomena through interactive and observable experiences, otherwise, this may become impossible. Accurate simulations enable the visualization of celestial mechanics, the exploration of the huge scales, and the observation of phenomena with different timescales [14][16][18]. There are some scientific principles about solar system simulations, different simulation methods, computational methods, and the tools available for scientific visualization, highlighting their relevance for educational applications.

To ensure the scientific accuracy and credibility of educational games, real astronomical data and physical simulations are required. This is achieved through the integration of several key components: applying principles such as Kepler's Laws and Newtonian Mechanics to predict planetary motion based on gravitational forces and orbits [21]; N-Body Simulations are utilized for modeling the complex gravitational interactions between multiple celestial bodies, to provide a dynamic representation of the system's evolution [22]; and incorporating astronomical data integration by collecting datasets from sources like NASA, CNSA, and other institutions [23]. These data sets typically include precise parameters like orbital parameters, masses, sizes, and surface characteristics of planets, moons, asteroids, and comets. Integrating these real-world data can make sure that the simulation environment can reflect the actual solar system as accurately as possible within the simulation.

Besides the fundamental orbital motion governed by gravity, an accurate solar system simulation for educational purposes must also involve key physical characteristics and motions of celestial bodies to provide a complete and realistic visualization. This includes simulating planetary rotation (spin around their axis), axial tilt (the angle between the planet's rotation axis and its orbital plane), and orbital inclination (the angle of a planet's orbital plane relative to a reference plane) [24]. Accurately representing these elements is essential for describing observable phenomena such as day-night cycles on different planets, the sun and stars in the sky, and the true three-dimensional arrangement of the solar system. Integrating these specific attributes, typically collected from astronomical data, would provide learners with a more complete and scientifically accurate way.

There are significant computational challenges for achieving high scientific accuracy

in solar system simulations, particularly for interactive or real-time applications like educational games [20]. Simulating the complex gravitational interactions of multiple bodies accurately over extended periods requires efficient algorithms, such as optimized variations of N-Body methods or hierarchical approaches, to maintain performance for smooth interaction. Developers often face the problem about the balance among achieving perfect scientific content, the need to ensure sufficient computational performance for real-time rendering and smooth user interaction, and the need to present complex phenomena in an understandable and engaging way for learners without oversimplifying to result in inaccuracy. Balancing the demands of scientific rigor with the limitations and requirements of interactive gameplay and educational design is a key consideration in developing effective educational simulations [22][25]. This usually involves which aspects are most critical to the educational goal to simulate, and which simplifications can be accepted without harming the learning experience. For example, accurately simulating the gravitational impact of each asteroid in the asteroid belt may be very difficult, but modeling the overall gravitational effect of the asteroid belt may be a reasonable option.

Several visualization tools are currently available that are related to scientific simulation for solar system education, they provide users interactive models of the universe. Some examples like Celestia, which is an open-source 3D astronomical simulation software, allow users to explore the universe, including the solar system, with a high degree of scientific accuracy. Stellarium is another famous astronomical simulation software that provides a realistic, real-time depiction of the night sky from any location on Earth. The simulation includes the accurate positioning and movement of the solar system bodies based on astronomical data and calculations. The visualization and observation are excellent, but the educational application of these tools often depends on external guidance or curricula. Also, some other tools such as SpaceEngine provide high-quality renderings of planets and galaxies based on scientific data, allowing for breathtaking visual exploration.

There are also some educational games and tools have been developed with the aim of teaching astronomical concepts, each possessing has strengths and limitations in engaging learners and conveying scientific knowledge. The representative existing solar system educational games and visualization tools will show their characteristics, so that to evaluate their effectiveness in facilitating learning, and identify areas for potential improvement, thereby providing valuable insights for the design of future educational tools in this domain. A review of current solar system educational games and visualization tools reveals a range of approaches to engaging users with astronomical content. There are some examples that highlight different design and educational concepts.

Universe Sandbox, an interactive physics simulation game, it serves as a powerful tool for demonstrating large-scale universe phenomena, such as planetary collisions, gravitational interactions, and stellar evolution. This game allows users to experiment with physical laws in an open-ended sandbox environment. It highly engaging for exploring principles of celestial mechanics through experimentation, but its primary strength lies in simulation and visualization rather than providing a structured curriculum or detailed educational content about individual solar system bodies, like their specific characteristics, or historical context. It primarily focuses on the exploration of physics principles rather than the

systematic acquisition of solar system knowledge (see Figure 2.1).

Although the open world of the Universe Sandbox allows for creative exploration and discovery, it can also be a problem to start for learners who lack a foundation in astronomical concepts or prefer a more guided learning experience. Additionally, as a game, it may lack structured objectives and may require a higher configuration to run smoothly.

Kerbal Space Program (KSP) is a complex spaceflight simulation game based on realistic orbital mechanics and rocketry principles, it does a great job in scientific accuracy in simulating physics relevant to space travel. Players learn hands-on by designing, building, and launching spacecraft to finish various missions within a simulated solar system, so that provides understanding of orbital dynamics and space engineering challenges. However, KSP has a steep learning curve that can present a significant barrier to entry for learners who are primarily interested in solar system astronomy rather than engineering or spaceflight simulation. Furthermore, it focuses on the process of spaceflight rather than providing comprehensive educational content about the solar system objects and phenomena (see Figure 2.2).

The KSP simulated solar system, while based on real-world data and physical properties, it is not a perfect copy of our solar system, and it may lack detailed information about the planets and moons themselves, which may lack educational value about the composition, history, and characteristics of the solar system. Its steep learning curve and spaceflight focus may not be suitable for all beginners who are interested in solar system astronomy.

NASA Eyes on the Solar System, which is developed by NASA. It is a robust 3D visualization software with real astronomical data to provide a scientifically accurate, real-time or simulated-time representation of celestial body positions, movements, and space missions. It is an invaluable resource for visualizing the scale and spatial arrangement of the solar system and tracking real-world astronomical events and missions. Nevertheless, as a visualization tool, it lacks the interactive challenges, explicit learning objectives, and reward systems inherent in educational game design to drive learning through active gameplay and structured progression. Its strength is in observation and visualization, not in interactive learning or knowledge assessment (see Figure 2.3).

NASA Eyes on the Solar System provides an accurate and stunning visual display of the solar system, but its passive nature may not popular as games that are more interactive and goal-oriented experiences. The lack of explicit educational content or goals, so if users can explore and learn effectively, they will need to have some basic knowledge and motivation.

These cases demonstrate various methods for accurate real-time visualization based on scientific data to engage users with solar system concepts through open-ended physics simulations and engineering challenges. They highlight the potential of interactive digital environments for astronomy education. Although the reviewed games and tools have valuable experiences for astronomy education, there are still some limitations, particularly for developing a comprehensive and accessible solar system educational game.

High scientific or gameplay entry barrier is a typical limitation. As seen with tools like KSP and Universe Sandbox, some existing platforms require learners to possess significant

fundamental knowledge of astronomy or physics principles, or a high level of experience with complex gameplay mechanics, so that players can fully understand and effectively engage with the educational content presented. This can limit the accessibility of these tools for novice learners.

Lack of structured learning paths is another limitation. Many existing games or tools, including open-ended simulations, they lack clear and task-driven learning objectives to guide players through a systematic acquisition of specific solar system knowledge. Learners may explore without clear educational goals, this will make it challenging to ensure comprehensive understanding of key concepts and facts [26].

Also, there is a limited integration of simulation, gameplay, and education. There is often a gap in seamlessly integrating scientifically accurate simulation with engaging gameplay mechanics and a clear and accessible educational story or content layout. Many existing tools may focus on one area, like simulation accuracy or visualization, but miss or ignore others, like game design.

Additionally, there should be less passive observation. Interactivity in some existing games is primarily restricted to navigation, changing perspectives, or operating display options. There are fewer opportunities for learners to engage in hands-on exploration, manipulate astronomical variables within a controlled environment, or participate in challenges directly to reinforce specific learning outcomes through active interaction with the simulated environment. Active learning allows learners to actively participate in the learning process through problem solving and experimentation, which is more effective than passive learning. Educational games should provide opportunities for active participation by allowing learners to actively observe the game environment, solve problems related to the solar system, and make decisions in the game world that will affect the direction of the game.

These limitations suggest that while valuable elements exist for various tools, a fully integrated and accessible educational game experience is still an area with significant space for development. This game should successfully merge the immersive potential of digital environments, the rigor of scientific simulation, and the motivational power of interactive gameplay, so that can provide a comprehensive, engaging, and accessible solar system learning experience. It should be accessible to a broad audience without extensive prior scientific knowledge. So that the shortcomings of the existing game would be fixed [27][28].

By integrating these elements, this project aims to provide an educational game that is both scientifically rigorous and engaging. It effectively combines entertainment and learning in astronomy education, as well as finds the balance between study and game playing. This game optimizes the learning curve to make the game more accessible for beginners. Also, while incorporating combat mechanics and a storyline to make the game world richer and immersive.

Figure 2.1: Universe Sandbox

Figure 2.2: Kerbal Space Program

Figure 2.3: NASA Eyes on the Solar System

3 Foundations

This chapter provides a theoretical foundation and necessary content background for the development of a solar system learning game. First, there is an introduction of the concept of Game-Based Learning (GBL), its core theories, and its applications and challenges in the field of education. Understanding the principles, effectiveness, and limitations of GBL is crucial for designing and developing an educational and engaging learning game. Secondly, this chapter will present a comprehensive overview of the game's theme, which is the solar system, including key knowledge points about the Sun, the eight major planets, dwarf planets, moons, asteroids, comets, and the structure and formation of the solar system. By combining the educational function of GBL with the scientific content of the solar system, this chapter lays the foundation for the detailed discussions on game design and implementation in the following sections.

3.1 Game-Based Learning (GBL)

Game-Based Learning (GBL), as an innovative educational approach, it is getting more attention. It integrates the interactivity of games into the teaching process, aiming to enhance students' engagement, motivation, and knowledge retention. This section will introduce various aspects of game-based learning and the foundation of the solar system, first explaining its definition, theoretical foundations, and examples of its application across different educational fields. Then, it will provide a detailed analysis of the advantages of GBL. Then, this chapter will objectively discuss the challenges faced in the implementation and promotion of game-based learning, in order to provide a comprehensive and deeper perspective. Finally, will introduce the basic knowledge of the solar system.

3.1.1 What is Game-Based Learning

Game-Based Learning (GBL) has gained significant attention as an effective educational tool. It enhances student engagement, motivation, and knowledge retention by interactive environments [7][8][29][30]. This section explores the theoretical foundations of GBL, its applications in different educational domains, and its advantages and challenges in effective learning experiences.

Game-Based Learning (GBL) is rooted in several key educational theories. The first theory is Constructivism, according to these theories, learners construct knowledge through active engagement, exploration, and experience rather than passive absorption of information. GBL aligns with constructivism by providing interactive learning environments where students experiment with concepts and learn through trial and error [31]. The

second is Self-Determination Theory (SDT), which emphasizes three psychological needs: autonomy, competence, and relatedness, which would drive motivation. Games fulfill these needs by offering choices, challenging tasks, and collaborative opportunities [32]. Also, the flow theory suggests that people achieve deep concentration and optimal learning when the level of challenge is balanced with their skill level. Well-designed educational games maintain this balance to keep learners engaged and prevent boredom or frustration [33].

GBL has been widely applied across various disciplines, including mathematics, language learning, medical training, and science education. For example, mathematics education games like DragonBox (see Figure 3.1 left) and Prodigy integrate mathematical problem-solving into engaging narratives, helping students develop numerical skill through interactive gameplay. Language learning Applications such as Duolingo (see Figure 3.1 right) and Rosetta Stone utilize gamification elements like rewards, progress tracking, and adaptive difficulty to reinforce vocabulary and grammar study. Science education simulations such as SimVenture for medical training, Driving Licence Simulation (see Figure 3.2), and Moonbase Alpha by NASA offer interactive environments where learners engage in realistic problem-solving scenarios.

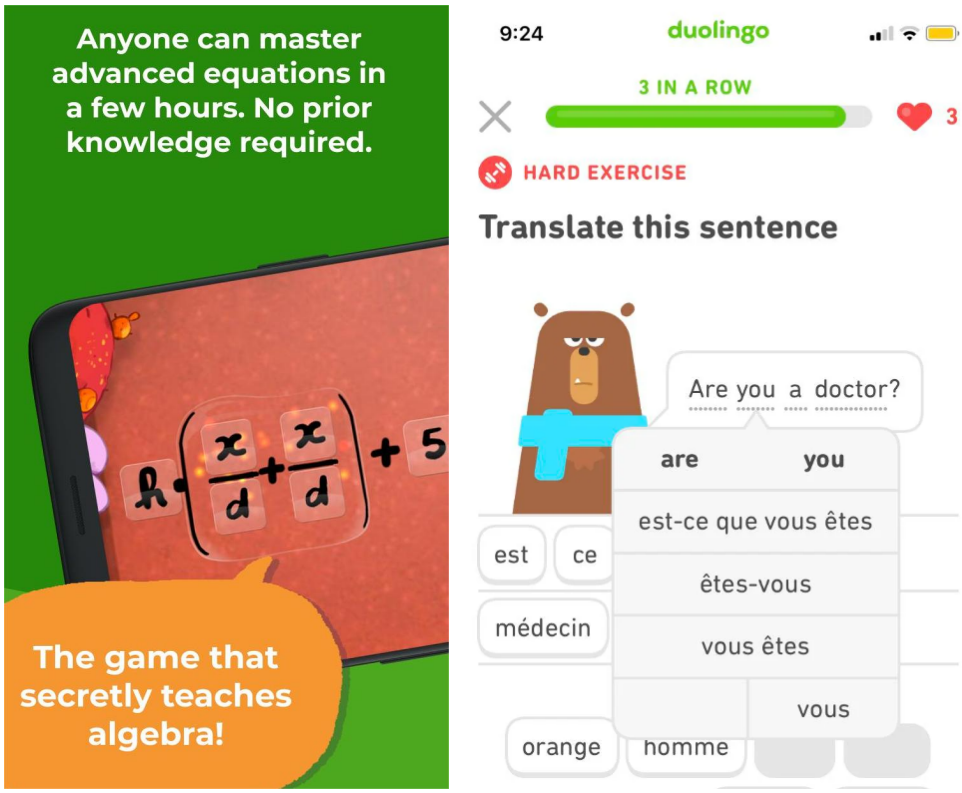

Figure 3.1: DragonBox(left) and Duolingo (right)

Figure 3.2: Driving Licence Simulation

3.1.2 Advantages of Game-Based Learning

Integrating gamified learning into science education offers various benefits. Firstly, it can significantly enhance student engagement and learning enjoyment. Through their fun and interactive nature, games can capture students' attention and excite their interest in learning, particularly for those who struggle with traditional teaching methods such as reading textbooks. Games also improve students' emotions, enhancing learning outcomes by attracting their attention and promoting active participation. Research has shown that GBL positively impacts both academic achievement and engagement in primary and secondary education [5][34][35].

Secondly, GBL promotes active participation. Unlike passively receiving information, games require students to take an active role in the learning process, making choices and influencing the game's progress. When learners are actively involved, they ask questions, seek information, and make connections, all of these are critical skills for deep learning [34].

Thirdly, GBL provides students with a safe environment for failure and experimentation. In games, students can freely try different approaches and learn from their mistakes without facing real-world consequences. This helps to encourage them to keep trying until they succeed or exit. Many games offer multiple attempts or built-in checkpoints, they allow students to learn from mistakes without restarting the whole game. This low-risk environment encourages students to explore various solutions to scientific problems, which can help them build resilience and develop a growth mindset [34][36].

Fourth, GBL offers immediate feedback. Games typically provide instant feedback to players, so they can quickly identify and correct mistakes. This continuous feedback loop keeps students engaged, allows them to adjust their strategies, and enhances their

understanding of scientific principles. Educational games provide assessment mechanisms as well, could help students to track their progress and performance in real-time, recognize their strengths and weaknesses, and make their understanding deeper [34][35].

Fifth, GBL enhances students' problem-solving and critical thinking skills. Many games require players to think critically and solve problems to advance, helping to develop valuable problem-solving skills. Science-based games often present increasingly complex problems that require students to find solutions by analyzing information based on their knowledge [34][37].

Sixth, GBL improves memory and knowledge retention. Unlike passive learning methods, games require active participation, engaging students through making choices, solving problems, and exploring the game world, thus helping them connect with the material. This active involvement reinforces neural pathways and promotes better information retention. Digital games as learning environments have been shown to support young people's social, emotional, and cognitive development, and evidence suggests they have a positive impact on long-term knowledge retention [34][37].

Finally, rigorously designed GBL strategies aligned with curricula can improve science test scores through fun and engaging learning. GBL fosters better attention, stronger memory, and problem-solving abilities, all of these are key elements for getting success in science education. For example, a study by WestEd showed that students who extensively used legends of learning games got significant improvements in their annual science test scores. A meta-analysis of data from more than 120 previous studies found that students learning STEM subjects through educational games consistently outperformed those learning through traditional methods [34][38].

3.1.3 Challenges of Game-Based Learning

The implementation and promotion of GBL also face a series of significant challenges, such as balancing education and entertainment, development costs, and the assessment and measurement of learning outcomes.

Finding the balance between the fun of educational games and their learning content is a core challenge. Research shows that an excessive focus on game mechanics or entertainment elements may cause players to become immersed in gameplay rather than educational objectives, which will negatively affect learning [20][39]. Successful design requires a good integration of learning objectives with gameplay, ensuring that the learning process itself drives the game's progression, and the entertainment should serve the educational purpose rather than replace it.

The development of high-quality educational games often requires significant resource costs. This involves the collaboration of different professional teams, including education experts, game designers, programmers, and artists, as well as investment in both software and hardware. Complex game design, high-quality audiovisual elements, and platform compatibility can all increase development costs, they also cause a developing barriers for many educational institutions or developers with limited budgets [40].

Accurately measuring the effectiveness of GBL and learners' actual learning outcomes is also a challenge. The non-linearity, interactivity, and individual differences inherent in

game experiences make it difficult for traditional assessment methods to fully capture the complex dynamic changes that occur during the learning process [41]. Research is exploring suitable assessment approaches for game environments, such as in-game analytics and performance-based assessments, but these require more complex design and validation, and currently, there are no unified standards and tools for evaluation [42].

3.2 The Solar System

The Solar System formed about 4.6 billion years ago from the gravitational collapse of a molecular cloud. It went through stages such as the formation of a protoplanetary disk, planetary accretion, and the late heavy bombardment.

The Solar System is a gravitationally bound system centered around the Sun, consisting of the Sun, eight planets and their moons, dwarf planets, asteroids, comets, meteoroids, and interstellar dust [43].

In this project, there are two main objects created: the Sun, eight planets, and their large moons. In this game, not all moons are created, cause most moons are too small to observe. Therefore, there are the ten largest moons are created in this game, which are Ganymede, Titan, Callisto, Io, Moon, Europa, Triton, Titania, Rhea, and Oberon [24].

3.2.1 The Main Celestial Bodies

The Sun is the core of the Solar System, accounting for 99.86% of its total mass. It is a G-type main-sequence star (yellow dwarf) primarily composed of hydrogen (about 74%) and helium (about 24%). Through nuclear fusion, the Sun converts hydrogen into helium, releasing vast energy to provide light and heat to the Solar System [44]. In the future, the Sun will expand into a red giant in about 5 billion years, eventually becoming a white dwarf [45].

Planets are divided into terrestrial planets and gas giants based on their composition and location. Characteristics of terrestrial planets are: small size, high density, rocky structure, and little or no rings. Characteristics of gas planets are: large size, low density, composed mainly of hydrogen and helium, with rings and multiple moons [46].

Mercury is the closest planet to the Sun in the inner solar system, with extreme temperature variations and no atmosphere. The average distance is about 57.9 million kilometers. With a diameter of about 4,880 kilometers, it is the smallest planet in our solar system, only about one-third the size of Earth. Mercury has 5.5% of Earth's mass and only 38% of Earth's gravity. As an Earth-like planet, Mercury is composed of rocks and metals and is characterized by its high density and large core. Mercury rotates very slowly, with a rotation period of about 58.6 Earth days, while its revolution period is only 88 Earth days, making it the fastest rotating planet in the solar system. Due to the special resonance effect between Mercury's rotation and revolution, a day on Mercury (from sunrise to sunrise) is equivalent to about 176 Earth days. Mercury's atmosphere is extremely thin, known as the exosphere, and consists mainly of hydrogen, helium, oxygen, sodium and potassium. This prevents Mercury from being able to insulate itself effectively, resulting

in an extreme temperature difference between its daytime and nighttime temperatures. Daytime temperatures can reach as high as 430 C, while nighttime temperatures plummet to -180 C. Mercury's surface is riddled with craters, similar to those on the moon, showing a long history of impacts. Mercury's surface has many unique topographic features, such as cliffs and escarpments, which may be the result of cooling contractions in Mercury's interior. Notable impact basins include the Calorie Basin, which is 1,550 kilometers in diameter and was formed by an ancient giant impact [47]. Mercury has no natural moons [48].

Venus is the second closest planet to the Sun in the inner solar system, with an average distance of about 108 million kilometers. With a diameter of about 12,104 kilometers, it is so similar to Earth that it is often referred to as "Earth's sister planet". However, although it is similar in size and structure to Earth, Venus has a very different environment, full of extreme conditions. Venus has a very slow and specialized rotation, with a period of 243 Earth days, which is longer than its revolution period of 225 Earth days. In addition, Venus rotates in the opposite direction, which means it rotates from east to west, which causes the sun to rise in the west and set in the east of Venus. The atmosphere of Venus is extremely thick, consisting mainly of carbon dioxide and a small amount of nitrogen, and also contains dense clouds of sulfuric acid. This thick atmosphere leads to an extremely strong greenhouse effect, making Venus' surface temperature as high as 465 C, making it the hottest planet in the solar system, surpassing even Mercury, which is much closer to the Sun. The surface pressure of Venus is about 92 times that of the Earth, which is equivalent to being at the bottom of the ocean at a depth of 1,000 meters. Such a high-pressure and high-temperature environment makes Venus an extremely inhospitable world for life. The surface of Venus is covered by lava plains, volcanoes and mountains, and scientists have discovered more than 1,600 volcanoes, some of which may still be geologically active. Sulfuric acid clouds in Venus' atmosphere reflect large amounts of sunlight, making Venus one of the brightest objects in the night sky, second only to the Moon [47]. Venus has no natural moons [48].

Earth is the only known planet in the inner Solar System to support life, serving as the home of humanity. It lies within the habitable zone of the Solar System, approximately 150 million kilometers from the Sun, with an orbital period of 365.25 days and a rotation period of about 24 hours. Earth offers ideal conditions for life, including an oxygen-rich atmosphere, liquid water, moderate temperatures, and a strong magnetic field that shields it from solar winds. About 71% of Earth's surface is covered by oceans, while the remaining 29% consists of landmasses such as mountains, forests, deserts, and plains. The atmosphere is composed of 78% nitrogen and 21% oxygen, providing breathable air and protecting the planet from meteors and harmful radiation [47]. Earth has only one natural moon: the Moon [48].

Mars is also in the inner solar system, it has a diameter of about 6,792 kilometers, which is about half the size of the Earth. Its surface gravity is about 38% of Earth's, so the weight of objects on Mars is significantly reduced. Mars has a revolution period of about 687 Earth days and a rotation period of 24 hours and 39 minutes, which is very close to the Earth's rotation period. Mars is a red planet, and this red color comes from the

abundance of iron oxide on its surface. The atmosphere of Mars is very thin, consisting mainly of carbon dioxide (about 95%), nitrogen and argon. Due to its low atmospheric density, Mars is unable to retain heat efficiently, resulting in a huge temperature difference between day and night, with temperatures usually ranging from -140 C to 20 C. There are polar ice caps on Mars, which are composed of water ice and carbon dioxide ice. Scientists hypothesize that there was once liquid water on Mars, and that there may be a large amount of ice hidden under the surface, and that life may even have existed, so humans consider Mars to be a possible immigrant planet [47]. Mars has 2 natural moons: Phobos and Deimos [48].

Jupiter is the largest and most massive planet in the outer solar system, with a diameter of about 142,984 kilometers, or 11 times that of Earth, and a mass of about 318 times that of Earth. Jupiter is a gas giant planet, composed mainly of hydrogen and helium, with no solid surface. Its rapid rotation gives it a rotation period of only about 9 hours and 55 minutes, but its revolution period is 11.86 Earth years long. One of Jupiter's most notable features is its famous Great Red Spot, a huge anticyclonic storm that has existed for centuries and is larger than the size of Earth in diameter. Jupiter's magnetic field is extremely strong, and the magnetosphere extends far beyond the other planets in our solar system. Jupiter also has a faint but massive system of planetary rings composed of dust and tiny particles. Jupiter's immense gravitational pull plays an important role in influencing the entire solar system, and is especially critical in protecting the inner planets from asteroid and comet impacts [47]. Jupiter has 95 natural moons [48], there are four the biggest in this project: Io, Europa, Ganymede, and Callisto.

Saturn is the second largest planet in the outer solar system and is about 1,429 million kilometers from the Sun. Saturn has a diameter of about 120,536 kilometers, which is about nine times that of the Earth, and a mass of 95 times that of the Earth. It is a gas giant planet, composed mainly of hydrogen and helium. Saturn rotates rapidly, with a rotation period of about 10 hours and 33 minutes and a revolution period of 29.5 Earth years. Saturn's most striking feature is its spectacular system of planetary rings. Saturn's rings are composed of ice grains, dust and rock fragments and are hundreds of thousands of kilometers wide, but only about 10 meters thick. The rings are divided into several major ring regions, such as rings A, B, and C, which surround Saturn's equatorial plane and appear brilliant when reflecting sunlight. Saturn's ring system is the most complex and pronounced planetary ring in the solar system. Saturn's atmosphere has a bright yellowish appearance due to clouds formed by ammonia ice crystals. There is storm activity in its atmosphere, the most famous of which is the Arctic Hexagonal Storm, a mysterious hexagonal-structured storm with a diameter of 30,000 kilometers and wind speeds of up to 500 kilometers per hour [47]. Saturn has 83 natural moons [48], there are two the biggest in this project: Rhea and Titan.

Uranus is about 2.87 billion kilometers from the Sun, which is the second farthest planet from the Sun in the outer solar system. It has a diameter of about 50,724 kilometers, which is slightly larger than Neptune's and four times that of Earth. With a mass 14 times that of Earth, Uranus is classified as an ice-giant planet, composed primarily of hydrogen, helium, and large amounts of icy methane, ammonia, and water. Uranus' most

notable feature is that its axis of rotation is nearly parallel to the plane of its orbit, with a tilt angle of 97.8 degrees, meaning that Uranus rotates almost "sideways". This extreme rotational inclination causes the poles of Uranus to experience polar days and polar nights for up to 42 years. Uranus has a rotation period of about 17 hours and 14 minutes, and a revolution period of 84 Earth years. Uranus' atmosphere has a special light bluish-green color, which is due to the presence of methane in its atmosphere, which absorbs red light and reflects blue light. Uranus' atmosphere is extremely cold, with an average temperature of about -224 C, making it the coldest planet in the solar system. Although Uranus' atmosphere is relatively calm, scientists have observed occasional huge storms and cloud system activity on its surface. Uranus' internal structure consists of a rocky core, an icy mantle rich in water, ammonia, and methane, and a thick hydrogen-helium atmosphere. Because Uranus emits less internal heat, it does not have significant internal heat radiation like Jupiter and Saturn [47]. Uranus has 27 natural moons [48], there are two the biggest in this project: Titania and Oberon.

Neptune is the farthest planet from the Sun in the outer solar system, at a distance of about 4.5 billion kilometers. It has a diameter of about 49,244 kilometers, which is about four times that of the Earth, and a mass of 17 times that of the Earth. Neptune is a gas giant planet whose main constituents include hydrogen, helium and methane, with the methane in its atmosphere absorbing red light, giving Neptune a beautiful deep blue color. Neptune has a rotation period of 16 hours and 6 minutes and a long revolution period of about 165 Earth years. Because of its great distance from the Sun, Neptune's surface temperature is extremely low, averaging about -214 C. Despite this, Neptune is home to one of the strongest storm systems in the solar system, with atmospheric winds reaching speeds of up to 2,000 kilometers per hour, far faster than hurricane speeds on Earth. Notable storm systems on Neptune include the Great Black Spot, which is similar to Jupiter's Great Red Spot, and is a huge storm that lasts for years. Neptune's internal structure consists of a rocky core, a layer of icy material and a thick outer layer of gas. Scientists hypothesize that there may be "diamond showers" deep inside Neptune, where extreme pressure breaks down methane into carbon, creating diamond particles [47]. Neptune has 14 natural moons [48], there is the biggest one in this project: Triton.

3.2.2 Other Celestial Bodies

In addition to the Sun and the eight major planets, the universe also contains a wide variety of other celestial bodies. These objects are distributed across different regions of the solar system, such as dwarf planets, the Oort Cloud, comets, meteors, and meteorites.

Dwarf planets are planetary-mass objects in a planetary system that do not fully meet the criteria for classification as a planet, yet are not considered satellites. In the Solar System, dwarf planets orbit the Sun directly and have sufficient mass and gravity to achieve a near-spherical shape under hydrostatic equilibrium, but unlike the eight classical planets, they lack the gravitational dominance to clear their orbital neighborhood [49]. The main dwarf planets include Pluto, Eris, Haumea, and Makemake.

Pluto was classified as the ninth planet, but is now recognized as a dwarf planet in the Kuiper Belt. It has a thin atmosphere of nitrogen, methane, and carbon monoxide

and is known for its large moon, Charon. Eris is slightly smaller than Pluto. Eris is a distant dwarf planet in the scattered disk region of the Kuiper Belt. Its discovery in 2005 contributed to Pluto's reclassification. Haumea is unique for its elongated shape, likely caused by its rapid rotation. It also possesses a ring system and two moons, Hi'iaka and Namaka. Ceres is the only dwarf planet in the Asteroid Belt, Ceres is the largest object there. It has a rocky composition and a possible subsurface ocean. Makemake is another Kuiper Belt object. Makemake has a bright surface covered in methane ice and is one of the coldest known dwarf planets [49].

The Asteroid Belt is a region between the orbits of Mars and Jupiter, containing a vast number of rocky and metallic objects. These bodies vary in size from tiny pebbles to large asteroids [50]. Ceres is the largest and only dwarf planet in the asteroid belt. Vesta is one of the largest asteroids, Vesta has a differentiated interior and signs of past volcanic activity. Pallas and Hygiea are the other major asteroids, composed mostly of rock and metal. For example, the Kuiper Belt is a vast region beyond Neptune's orbit, filled with icy bodies and dwarf planets. Some notable objects include Pluto and Eris, it is the largest known Kuiper Belt objects. Haumea and Makemake are the other significant members of the belt. Quaoar and Sedna are large trans-Neptunian objects that may offer insights into the Solar System's formation [50].

Comets are icy bodies that originate from the Kuiper Belt or the Oort Cloud. When they approach the Sun, their volatile compounds vaporize, creating a glowing coma and tail. Famous examples include: Halley's Comet, which is a short-period comet that returns approximately every 76 years. Comet Hale-Bopp is one of the brightest comets observed in the 20th century. The Comet NEOWISE is a recently discovered comet visible to the naked eye in 2020 [51].

A meteoroid entering Earth's atmosphere burns up as a meteor, and if it survives to reach the ground, it is called a meteorite. Meteoroids are small rocky or metallic fragments in space, often originating from comets or asteroids. When a meteoroid enters Earth's atmosphere, it burns up due to friction, creating a bright streak known as a meteor. If a meteoroid survives its descent and lands on Earth's surface, it is called a meteorite. Meteorites provide valuable insights into the composition of celestial bodies [52].

The Solar System's extent is determined by the Sun's gravity and solar wind, with key boundaries. The heliopause is the outermost edge of the Sun's influence, where the solar wind slows down and merges with the interstellar medium. This boundary marks the transition from the Solar System to interstellar space. Voyager 1, launched in 1977, became the first human-made object to cross this boundary in 2012, followed by Voyager 2 in 2018 [53].

The Oort Cloud is the farthest region where the Sun's gravity can still hold objects in orbit. It is believed to be a vast, spherical shell of icy bodies, possibly extending from 1 to 2 light-years from the Sun. It serves as a reservoir for long-period comets that occasionally enter the inner Solar System when their orbits are disturbed by gravitational influences [54].

3.2.3 Human Exploration of the Solar System

Up to now, humanity has made significant strides in exploring the solar system through robotic missions and crewed spaceflights, such as China, the USA, and Europe.

The China Lunar Exploration Project (CLEP), also known as the Chang'e Project, is China's lunar exploration program initiated by the China National Space Administration (CNSA). It was officially approved on January 23, 2004. China has achieved world-leading advancements in the lunar exploration program, which is represented by the Chang'e program [55][56].

The project consists of three primary phases: The first is orbiting the Moon, which means launching lunar orbiters. The second is landing on the Moon, which means deploying unmanned rovers for surface exploration.The third phase is sample Return, sending probes to collect lunar samples and return them to Earth. The Chang'e program achieved multiple milestones, including the first soft landing on the Moon's far side (Chang'e 4) and lunar sample return (Chang'e 5) in the world [55][56].

The Planetary Exploration of China (PEC) is the planetary exploration program of the China National Space Administration (CNSA) for exploring the Solar System. The series of missions under this program is named "Tianwen" and was officially approved on January 11, 2016 [55][56].

The Tianwen-1 mission, which is the Mars exploration mission, launched in 2020, successfully placed an orbiter around Mars and landed the Zhurong rover, which conducted scientific studies of the Martian surface and atmosphere [55][56].

At the end of 2022, China completed the construction of the Tiangong Space Station. Tiangong is currently the only space station in long-term operation besides the International Space Station, and unlike the International Space Station, Tiangong was independently developed by China. The completion of the Tiangong Space Station marks another major milestone in China's manned spaceflight endeavors. China became one of the few countries that have a long-term operational space station, and some years later, China will be the only country that has an independent space station. Currently, the Shenzhou space ships are conducting hundreds of scientific experiments aboard the Tiangong Space Station, including research in microgravity. In the future, the Tiangong Space Station will be expanded further [55][57].

China has future plans for exploring the Solar System, including the moon, planets, and outer space, for example, the Fourth Phase of the Lunar Exploration Project "Survey": Initiated in 2021, it refers to the subsequent scientific activities on the Moon after completing the "three-step" process. This involves a comprehensive survey of the Moon's south pole, including investigations of its terrain, topography, material composition, and space environment. Additionally, it aims to verify certain technologies in preparation for the future lunar scientific base establishment [55][56].

On March 20, 2024, the Chang'e-2 relay satellite was successfully launched. As China's public relay satellite platform for the fourth phase of lunar exploration, the Chang'e-2 relay satellite will provide communication relay services for the Chang'e-4, 6, 7, and 8 missions, and the construction of the basic model of the lunar scientific research base. Among them, the Chang'e-6 mission has already been launched on May 3, 2024, and its

return capsule carried a total of 1935.3 grams of lunar soil samples from the far side of the Moon back to Earth [56]. Also, there will be more planetary exploration in the future, for example: Tianwen-2 will explore asteroids and comets, Tianwen-3 will explore Mars, and Tianwen-4 will explore Jupiter [56].

The Apollo program was a series of manned space missions conducted by NASA from 1961 to 1972. During the 1960s, its main goal was to achieve manned lunar landings and safely return to Earth. The missions of the "Apollo" spacecraft included preparations for manned lunar flights and the realization of such missions, which concluded by the end of 1972 [58][59]. The Apollo program provided detailed insights into the lunar surface's characteristics, chemical composition, optical properties, and also explored the Moon's gravity, magnetic field, and lunar seismic activity [58][59].

The United States is at the forefront of planetary exploration, with significant missions, represented by the Flagship Program, which is a series of NASA's solar system exploration missions. The Viking Program captured high-resolution images of the Martian surface, analyzed atmospheric composition, and searched for evidence of life on Mars. Explored Jupiter, Saturn, Uranus, Neptune, and interstellar space, providing valuable data on gas giants [59]. The Galileo spacecraft was the first to orbit Jupiter and conduct atmospheric studies of the planet. Galileo's research indicated the presence of a subsurface ocean beneath Europa's ice crust. It discovered the first known moon of an asteroid [59]. Cassini-Huygens investigated Saturn, its rings, and landed on its moon Titan. It is the first probe to successfully land on an outer solar system body [59]. Mars Science Laboratory and Mars 2020 examine Mars' past and present habitability and assess its long-term geological evolution. Perseverance determines whether life ever existed on Mars and collects rock and soil samples. Ingenuity demonstrates powered flight feasibility in Mars' thin atmosphere [59]. The Europa Clipper (formerly known as the "Europa Multiple Flyby Mission") is an ongoing NASA interplanetary orbiter mission. Launched on October 14, 2024, its primary objective is to continue the research initiated by the Galileo spacecraft by conducting a series of flybys to study Europa, one of Jupiter's Galilean moons, while orbiting Jupiter [59].

There are also some other missions like: Juno Mission, which is the second spacecraft to enter Jupiter's orbit. It was launched from Cape Canaveral Air Force Station on August 5, 2011, and entered Jupiter's polar orbit on July 5, 2016. The probe is positioned in a polar orbit to study Jupiter's composition, gravity field, magnetic field, magnetosphere, and poles. Voyager Missions, which were launched in 1977, are the first probes to provide detailed images of Jupiter, Saturn, and their moons. It is the farthest artificial spacecraft from Earth in history and the artificial object to leave the solar system [59].

The Hubble Space Telescope has had a transformative impact on astronomy, providing stunning images and invaluable data that have advanced numerous fields of astrophysics. The James Webb Space Telescope, as the next-generation flagship observatory, offers powerful infrared observation capabilities, enabling it to observe the early universe and study the formation of stars and planetary systems. Other significant space telescopes include the Chandra X-ray Observatory and the Spitzer Space Telescope, which observe the universe in different bands of the electromagnetic spectrum, offering a more comprehensive

view of the cosmos [58][59].

The USA also has its future exploration plan for the Solar System. For example, the Artemis lunar program is an international cooperative effort to explore the Moon, led by NASA, and officially launched in 2017. It involves six collaborating agencies: the European Space Agency (ESA), Japan's JAXA, the German Aerospace Center (DLR), the Italian Space Agency (ASI), the Israel Space Agency (ISA), and the Canadian Space Agency (CSA). The Artemis program aims to reestablish a human presence on the Moon for the first time since the Apollo 17 mission in 1972. The long-term goal of the program is to establish a permanent base on the Moon and facilitate human missions to Mars. The Mars Sample Return Mission, which is a collaborative plan between ESA and NASA to return Martian soil samples to Earth, assists in sample retrieval and return logistics [59].

While ESA has not conducted independent crewed lunar missions, it has contributed to global lunar exploration efforts. SMART-1 is the ESA's first lunar probe, which used ion propulsion technology to orbit the Moon and study its surface composition before intentionally crashing into it at the end of its mission [60].

ESA has successfully launched multiple planetary exploration missions. For example, Mars Express is the first Mars mission of ESA, which has been studying the planet's surface, atmosphere, and signs of liquid water for over two decades. The ExoMars Program, it is a joint mission with Roscosmos aiming to explore Mars and search for signs of life. Currently, ExoMars Trace Gas Orbiter (TGO) is analyzing Mars' atmosphere and mapping methane sources [60].

ESA continues to expand its role in space exploration, participating in international projects and developing independent missions to explore the Solar System. Argonaut (European Large Logistic Lander) is ESA's future lunar lander, designed to deliver cargo and equipment to the Moon to support the Artemis program and future lunar bases. ExoMars Rosalind Franklin Rover is a future plan of the ExoMars, planned to land and drill into Mars' surface to search for ancient microbial life. A Venus exploration mission, EnVision, is designed to study the planet's geology, atmosphere, and potential past habitability [59][60].

4 Teaching The Solar System With a Computer Game

This section introduces the core design elements of the game, including its name, gameplay, user interface, physics mechanisms, and educational system. The game aims to provide an engaging and scientifically accurate experience that seamlessly integrates learning and interactive exploration.

4.1 Game Overview

This subsection provides an overview of the game, including its name, background, and core objectives. The game is designed as an immersive educational experience that combines astronomy learning with an engaging storyline and interactive gameplay.

This game is named Wentian (Chinese: 问天), which originates from Chinese culture and symbolizes humanity's effort of cosmic mysteries. There are several main references: Tianwen, which means Heavenly Asking, is an ancient poem by Qu Yuan that raises deep questions about the origins of the universe and natural phenomena, reflecting the spirit of inquiry. This is the origin of the name Wentian [61]. Wentian also means to explore the space, as the title of a solar system educational game, it represents the player's journey in exploring the universe and knowledge learning. This name conveys a strong sense of technology, discovery, and the future, aligning well with the theme of astronomy education and scientific exploration. One of the laboratory cabin modules of the Tiangong Space Station is also named "Wentian".

This solar system educational game is designed to provide an immersive interactive experience, allowing players to learn astronomy while exploring the solar system with a vector engine-driven space fighter. The game adopts a mission-driven approach to guide players in gradually mastering the basic structure of the solar system, planetary characteristics, orbital motion, and astrophysical principles.

Unlike atmospheric flight, it is operated based on aerodynamics, space flight in this game is operated in a vacuum environment. The space fighter is controlled by a player from a first-person cockpit perspective, and the fighter has vector engines, which allow for directional control, which uses the vector force rather than aerodynamic lift or drag [62][63].

The game is set in a future war where the solar system faces an unprecedented threat. A mysterious group of alien invaders is attempting to plunder the precious resources that humanity depends on for survival. If they succeed, the future of humankind will be in grave danger. The player's Missions include: Pilot a spacefighter, protect the resources of

the solar system, repel the invaders, and safeguard humanity's hope.

4.2 Core Gameplay

This section outlines the core gameplay mechanics that shape the player's experience. The game features multiple modes designed to provide a balance between free exploration, structured missions, and challenges based on real scientific principles.

4.2.1 Game Mode

This game is designed as a Flight Combat Simulator (FCS) serious game. Players will pilot the space fighter to finish a series of missions, during this, players will also learn some knowledge about the solar system. It is a combination of game playing and studying, which makes the game meaningful as well as makes studying less boring.

The game includes the following main modules:

- **Exploration module:** Players can freely fly, explore, and observe different celestial bodies in the solar system.
- **Mission module:** Players complete a series of tasks based on real astronomical knowledge, such as planetary exploration, fighting enemies.
- **Challenge module:** Players must finish specific astronomical exploration or flight challenges to complete the game story.

The mission system is the core of the game, with each mission linked to a specific astronomical concept. Mainly includes two categories: One is the fighting mission and planetary exploration mission.

For the fighting mission, the player needs to pilot the space fighter to fight with enemies, so that protect the resources of the solar system. There are two types of combat missions in the game. One involves enemies attacking the player, where the player must eliminate all enemies to survive. The other involves enemies attacking a resource-delivering space cruiser, and the player's task is to destroy the enemies to protect the cruiser. There is a fighting system that includes an enemy indicator UI to trace enemies' positions, and a crosshair to help players aim enemies, and attack enemies with a laser cannon. It can be seen in Figure 4.1.

For the planetary exploration mission, the player will travel to specific planets, analyze their history, components, surface characteristics, and planetary movement. As players complete missions, they will acquire astronomical knowledge and can continually explore space freely for repeated study and gaming, or exit the game.

*9

Figure 4.1: Fighting

4.2.2 Game Controls

This is a First-Person Cockpit Mode: Players can control their fighter from a cockpit perspective for spaceflight in a virtual solar system, which is an immersive simulation game. This game is controlled by keyboard and mouse, where the keyboard is used for fighter power control, fighter stabilization, teleport, and shooting, and the mouse is used for the fighter's direction (camera rotation) control. Mouse to control the fighter's orientation by simulating a joystick with the mouse. Moving the mouse forward and backward controls the fighter's pitch (nose up and down), while moving the mouse left and right controls the fighter's yaw (turning left and right) (see Listing 4.1).

```
1 float pitchInput = Input.GetAxis("Mouse Y") * turnSpeed * Time.deltaTime;
2 float yawInput = Input.GetAxis("Mouse X") * turnSpeed * Time.deltaTime;
3
```

Listing 4.1: Mouse Input Code

Key-E to turn on or turn off the vector engine. When the engine is activated, the fighter accelerates; otherwise, the fighter will keep the current speed due to inertia. Additionally, there is a speed-dependent acceleration, which is different according to the current speed. Fighter accelerates faster at low speeds, and the acceleration gradually decreases as it approaches the maximum speed (see Listing 4.2).

```
if (Input.GetKeyDown(KeyCode.E)){
    isEngineOn = !isEngineOn;
}
if (isEngineOn) {
    float acceleration = baseAccelerationRate * (1 - (movementSpeed /
    maxSpeed));
    movementSpeed += acceleration * Time.deltaTime;
}

```

Listing 4.2: Acceleration Code

Key-S to activate engine reverse thrust. If the engine acceleration is turned off, players could activate engine reverse thrust by pressing `S`. If the current speed reaches zero, it means there is no forward acceleration anymore. If the engine reverse thrust is still activated, the fighter will go backward (see Listing 4.3).

```
if (Input.GetKey(KeyCode.S) && isEngineOn == false){
    movementSpeed -= baseDecelerationRate * Time.deltaTime;
}

```

Listing 4.3: Engine Reverse Thrust Code

Key-SPACE to shoot a laser. There is a cooldown timer management for shooting, in case of unlimited firing (see Listing 4.4).

```
if (Input.GetKey(KeyCode.Space) && Time.time >= nextFireTime){
    GameObject bullet = Instantiate(laserPrefab, firePoint.position,
    firePoint.rotation);
    nextFireTime = Time.time + fireRate; // Set next allowed fire time
    shootAudioSource.PlayOneShot(shootSound);
}

```

Listing 4.4: Shooting Code

Key-R to stabilize the fighter. If players get crashed or shot, they may lose control, so a stabilization function is needed to fix it. This function is considered for gaming balance (see Listing 4.5).

Key-SHIFT to activate super boost. The super boost will speed movement up five times, it used for decreasing the game time. If the engine is on and the `SHIFT` is pressed, super boost will be activated and deactivated when released. In case of unlimited boosting, there is a cooldown timer as well (see Listing 4.6).

```
if (Input.GetKeyDown(KeyCode.R)){
    isStabilizing = true;
    targetRotation = transform.rotation;
}
if (isStabilizing){
    // Smoothly reduce angular velocity to zero
    rb.angularVelocity = Vector3.Lerp(rb.angularVelocity, Vector3.zero,
    stabilizationSpeed * Time.deltaTime);
    // Gradually reduce rotation difference from the targetRotation
    transform.rotation = Quaternion.Slerp(transform.rotation,
    targetRotation, stabilizationSpeed * Time.deltaTime);
    // Stop stabilizing
    if (rb.angularVelocity.magnitude < 0.01f && Quaternion.Angle(
    transform.rotation, targetRotation) < 0.1f) {
        isStabilizing = false;
    }
}

```

Listing 4.5: Stabilization Code

```
if (Input.GetKeyDown(KeyCode.LeftShift) && !Input.GetKey(KeyCode.S) && !
    isBoostCooldown && !isBoostActive && isEngineOn && movementSpeed >= 0)
    {
    isBoostActive = true;
}
if (Input.GetKeyUp(KeyCode.LeftShift) && isBoostActive){
    isBoostActive = false;
    isBoostCooldown = true;
    boostCooldownTimer = boostCooldownDuration;
}
if (isBoostActive){
    // Smoothly increase the boost factor
    currentBoostFactor = Mathf.Lerp(currentBoostFactor, boostMultiplier,
    boostTransitionSpeed * Time.deltaTime);
}

```

Listing 4.6: Super Boost

Key-NUMBER to teleport the fighter to each planet. The numbers 1-8 correspond to the positions of the planets, and 9 corresponds to the position of the enemy space station. This function is also used for decreasing game time (see Listing 4.7).

```
Vector3 offset = (transform.position - planet.position).normalized *
    teleportDistance;
transform.position = planet.position + offset;

```

Listing 4.7: Teleport Code

4.3 User Interface (UI)

The user interface is an important component of a game, it provides the information display function. In this project, the UI is designed to enhance usability, engagement, and educational effectiveness. This section explores the key UI components in the game. The game's UI adopts a modular design to ensure clear and intuitive information presentation. Key UI components include several parts. There is a main panel, which displays an overview of the solar system, activated objectives, and current lighting.

4.3.1 Cockpit Head Up Display (HUD)

There are two main parts of HUD, including the radar and the technical information display. The radar displays the position of the player and planets, as well as direction. Technical information provides detailed technical information about the fighting status by Head Up Display (HUD), such as health point (HP), speed, yaw, pitch. The cockpit UI overview can be seen in Figure 4.2.

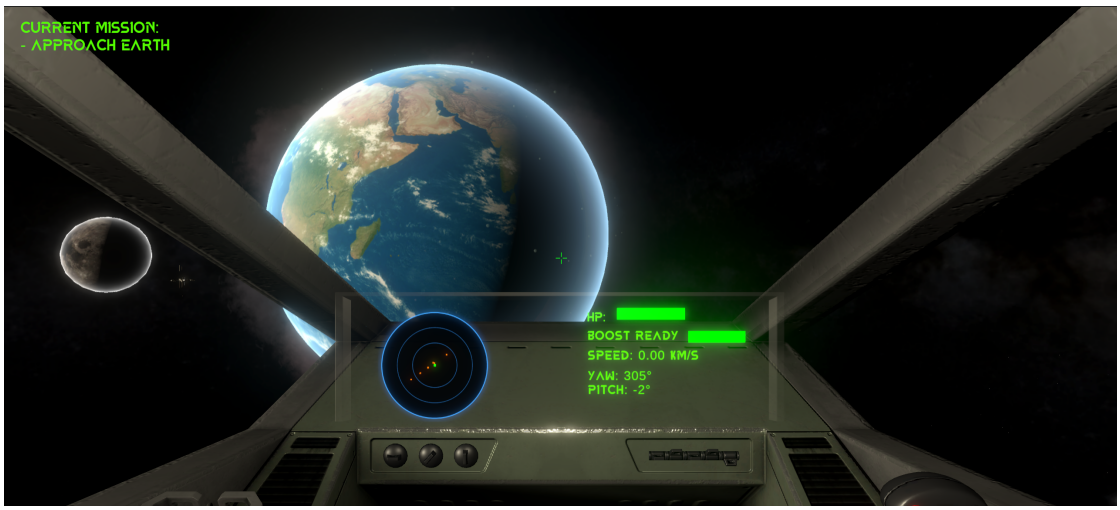

Figure 4.2: Cockpit

The radar system is a 2D UI image element, which is used for displaying the positions of objects around the player. It provides 3D spatial reference by displaying targets with a

colour range (from red to yellow) based on their vertical distance related to the player. This system is implemented using Unity's UI components and a custom C# script.

A background image provides the radar background (see Listing 4.8). It serves as the radar's visual canvas and coordinate reference frame and is configured as the parent object for all radar elements. It acts as the bounded coordinate space and provides the clipping boundary for out-of-range targets.

```
public RectTransform radarBackground;

```

Listing 4.8: Radar Backgoround

A player icon provides a fixed marker representing the player's position (see Listing 4.9). Firstly, instantiated as a child of the background, and create a 5×5 pixel green dot for visibility. It is always set at the center (0,0) in radar space.

```
void CreatePlayerIcon() {
    playerIcon = new GameObject("PlayerIcon");
    playerIcon.transform.SetParent(radarBackground, false);
    Image iconImage = playerIcon.AddComponent<Image>();
    iconImage.color = Color.green; // Constant player color
    iconImage.rectTransform.sizeDelta = new Vector2(5, 5);
    iconImage.rectTransform.anchoredPosition = Vector2.zero;
}

```

Listing 4.9: Player Icon

The radar dots represent various dynamic planets' positions within the radar's range. To implement this, the radar display radius is set as 30000, and created an object pool of 3×3 pixel dots. Firstly, creating the dots by initial coordinates of planets, and tracing planets' coordinates dynamically. Then calculating the position on radar via polar projection by converting world space to local XZ plane and applying rotational compensation for player yaw, which means radar direction will synchronize with the player's yaw angle (see Listing 4.10).

There is a height-based color coding to handle the object's vertical position display. The problem is: the space is a 3D environment, but the objects should be displayed in a 2D map, so a color buffer is used for displaying the y-position. Targets are colored from red (below the player) to yellow (above the player) based on their vertical distance. The color space interpolation and normalized vertical range (±10,000 units) are used for solving this (see Listing 4.11).

```
void CreateRadarDots(){
    foreach (GameObject target in targetObjects){
        GameObject radarDot = new GameObject("RadarDot");
        radarDot.transform.SetParent(radarBackground, false);
        Image dotImage = radarDot.AddComponent<Image>();
        dotImage.color = Color.red;
        dotImage.rectTransform.sizeDelta = new Vector2(3, 3);
        radarDot.SetActive(false);
        radarIcons.Add(radarDot);
    }
}
void Update(){
    Vector3 offset = target.transform.position - player.transform.
    position;
    float dist = new Vector2(offset.x, offset.z).magnitude;
    if (dist <= radarDisplayRegion) {
        Vector2 radarPos = new Vector2(offset.x, offset.z);
        radarPos = RotateVector2(radarPos, playerYaw);
        radarPos = radarPos.normalized * radarSize * (dist /
    radarDisplayRegion);
        radarDot.GetComponent<RectTransform>().anchoredPosition =
    radarPos;
    }
    radarDot.SetActive(dist <= radarDisplayRegion);
}
Vector2 RotateVector2(Vector2 v, float degrees){
    float radians = degrees * Mathf.Deg2Rad;
    float sin = Mathf.Sin(radians);
    float cos = Mathf.Cos(radians);
    float tx = v.x;
    float ty = v.y;
    v.x = (cos * tx) - (sin * ty);
    v.y = (sin * tx) + (cos * ty);
    return v;
}

```

Listing 4.10: Core Algorithm

```
float heightDifference = offset.y;
float colorFactor = Mathf.InverseLerp(-maxVerticalDistance,
    maxVerticalDistance, heightDifference);
Color targetColor = Color.Lerp(Color.red, Color.yellow, colorFactor);

```

Listing 4.11: Radar Dots Color Buffer

The UI Information System provides real-time feedback to the player by displaying essential game data such as speed, orientation, health points (HP), and boost status. This system ensures the player has the necessary data to check and manage, also make

decisions effectively.

There is a Health Point (HP) monitoring. The HP is the base for the fighter, which has a range from 0 to 100, in which 100 is the max HP, and if HP reaches 0, then the fighter is destroyed and game over. There is a color buffer to dynamically monitor the HP changing from red (0%) to green (100%). At the same time, create a color bar UI Image with `Image.Type = Filled` to display it on HUD (see Figure 4.4 and Listing 4.12).

```
float hpRatio = Mathf.Clamp01(playerControlScript.HP / (float)
    playerControlScript.maxHP); // Mathf.Clamp01 prevents overflow/
    underflow
HPFillImage.fillAmount = hpRatio;
HPFillImage.color = Color.Lerp(Color.red, Color.green, hpRatio);
HPText.text = "HP: ";

```

Listing 4.12: HP Color Buffer

To enhance the game's sustainability, an automatic HP recovery feature is added. If the player does not take damage for three seconds, the HP will automatically recover to the maximum value (see Listing 4.13).

```
timeSinceLastDamage += Time.deltaTime;
if (timeSinceLastDamage >= damageCooldown && HP < maxHP)
{
    HP = Mathf.Min(HP + hpRecoveryRate * Time.deltaTime, maxHP);
}

```

Listing 4.13: HP Recoverying

For boosting function, there is a cooldown management with three seconds. A visual feedback for boosting ability state is also needed. The time calculation is made by inverse progress (remaining time). The implementation is similar as the HP monitor, there is a color buffer to dynamically monitor the timer. Then create a color bar UI Image with `Image.Type = Filled` to display it on the HUD (see Figure 4.4 and Listing 4.14). To display the current movement speed by the speed setting in `playerControl` script (see Listing 4.15).

For flight, there should be a reference for flight orientation, but in this project, there is no rolling function, so just Yaw and Pitch are needed. Pitch is converted from [0,360] to [-180,180] range, and Yaw is displayed as the original 0-360° value. Additionally, there is a sign inversion for intuitive pitch display (positive = nose up) (see Listing 4.16).

```
if (playerControlScript.isBoostCooldown) {
    float progress = 1 - (playerControlScript.boostCooldownTimer /
    playerControlScript.boostCooldownDuration);
    boostFillImage.fillAmount = progress;
    boostFillImage.color = Color.Lerp(Color.red, Color.green, progress);
    boostCooldownText.text = "Charging";
}
else {
    boostFillImage.fillAmount = 1f;
    boostFillImage.color = Color.green;
    boostCooldownText.text = "Boost Ready";
}

```

Listing 4.14: Boost Color Buffer

```
speedText.text = "Speed: " + playerControlScript.realTimeSpeed.ToString("
    F2") + " km/s";

```

Listing 4.15: Speed Display

Figure 4.3: HUD Overview

```
float pitch = flighterTransform.eulerAngles.x > 180 ? flighterTransform.
    eulerAngles.x - 360 : flighterTransform.eulerAngles.x;
orientationText.text = "Yaw: " +
    flighterTransform.eulerAngles.y.ToString("F0") + "ř  \n" + "Pitch: " +
    (-pitch).ToString("F0");

```

Listing 4.16: Orientation Display

Figure 4.4: HUD When Recovering

4.3.2 Fighting UI

The complete combat UI consists of enemy indicators and a crosshair. A crosshair is a UI graphic located at the center of the screen, used to indicate the current aiming point. The Enemy Indicator System is an UI component for providing visual feedback to display enemy positions. This system enhances spatial awareness by displaying indicators when enemies are either off-screen or at a specific distance in-screen.

The `indicatorPrefab` is a simple Image component representing a visual marker for enemies. It is instantiated dynamically for each enemy object and managed through a dictionary to ensure efficient lookups and updates. Listing 4.17 shows the indicator creation logic.

```
public Image indicatorPrefab; // Indicator prefab (only Image)
private void CreateIndicatorForEnemy(GameObject enemy){
    // Create a new Image
    Image indicator = Instantiate(indicatorPrefab, transform);
    enemyIndicators[enemy] = indicator;
}

```

Listing 4.17: Indicator Creation

The position of each indicator is calculated in the `UpdateIndicatorPositionAndColor()` method. The system first determines the screen space position of the enemy using `Camera.WorldToScreenPoint()`. If the enemy is outside the screen boundaries or behind the camera, a clamping mechanism is applied to ensure the indicator remains visible along

the screen edges. The enemy indicator can be seen in Figure 4.5, and the positioning logic can be seen in Listing 4.18.

```
private void UpdateIndicatorPositionAndColor(GameObject enemy, Image
    indicator){
    Vector3 screenPos = mainCamera.WorldToScreenPoint(enemy.transform.
    position);
    Vector3 screenCenter = new Vector3(Screen.width / 2f, Screen.height /
     2f, 0);
    Vector3 direction = screenPos - screenCenter;
    if (screenPos.z < 0){
        direction.x = -direction.x;
        direction.y = -direction.y;
        screenPos *= -1;
    }
    bool isOffScreen = screenPos.x < 0 || screenPos.x > Screen.width ||
    screenPos.y < 0 || screenPos.y > Screen.height;
    if (isOffScreen || screenPos.z < 0){
        float angle = Mathf.Atan2(direction.y, direction.x);
        float screenWidthHalf = Screen.width / 2f - screenEdgeOffset;
        float screenHeightHalf = Screen.height / 2f - screenEdgeOffset;
        float x = Mathf.Cos(angle) * screenWidthHalf;
        float y = Mathf.Sin(angle) * screenHeightHalf;
        if (Mathf.Abs(x) > screenWidthHalf){
            x = Mathf.Sign(x) * screenWidthHalf;
            y = x * Mathf.Tan(angle);
        }
        if (Mathf.Abs(y) > screenHeightHalf){
            y = Mathf.Sign(y) * screenHeightHalf;
            x = y / Mathf.Tan(angle);
        }
        Vector3 edgePosition = screenCenter + new Vector3(x, y, 0);
        indicator.rectTransform.position = edgePosition;
    }else{
        indicator.rectTransform.position = screenPos;
    }
}

```

Listing 4.18: Indicator Positioning

If an enemy is close (within 80 units) or far (over 1000 units) enough to the player and is not off-screen, the indicator will be semi-transparent to reduce visual clutter. Additionally, if the enemy is directly in front of the player (within a 0.5-degree angle), which means the player is almost aiming at the enemy, the indicator is also semi-transparent. Additionally, the indicator color changes based on the enemy's position relative to the player's camera. Two colors are defined: `onScreenColor` (light red) and `offScreenColor` (dark red). The system uses a `Vector3.Dot()` calculation to determine if an enemy is in front or behind the player and interpolates the indicator color according to that, and then applies the alpha value (see Listing 4.19).

Figure 4.5: Enemy Indicator UI

```
// Calculate angle with camera forward direction
Vector3 toEnemy = (enemy.transform.position - mainCamera.transform.
    position).normalized;
float angleToEnemy = Vector3.Angle(mainCamera.transform.forward, toEnemy)
    ;
float distanceToEnemy = Vector3.Distance(mainCamera.transform.position,
    enemy.transform.position); // Calculate the distance to the enemy
float alpha = 1f;
// Make it semi-transparent if too close or aiming
if ((distanceToEnemy <= 80f || distanceToEnemy>= 1000f) && !isOffScreen)
    alpha = 0.3f;
if (angleToEnemy <= 0.5f)
    alpha = Mathf.Min(alpha, 0.3f);
UpdateIndicatorColor(enemy, indicator, alpha);
private void UpdateIndicatorColor(GameObject enemy, Image indicator,
    float alpha) {
    Vector3 toEnemy = (enemy.transform.position - mainCamera.transform.
    position).normalized;
    float dotProduct = Vector3.Dot(mainCamera.transform.forward,toEnemy);
    float angleFactor = Mathf.InverseLerp(-1f, 1f, dotProduct);
    Color targetColor = Color.Lerp(offScreenColor, onScreenColor,
    angleFactor);
    targetColor.a = alpha; // Apply custom alpha
    indicator.color = targetColor;
}

```

Listing 4.19: Indicator Setting

4.3.3 Mission UI

In this game, there is a mission UI that is the core interface for displaying and managing the player's current tasks and introducing the current celestial body. It consists of the task list and the introductions.

The mission list UI allows players to fast check the task status and progress, it also provides access to detailed task information. The UI related figures show how the mission list looks like. The task list can reflect the task progress in real-time like remaining enemies, and displaying multiple active tasks, removing completed tasks, and unlocking new tasks. Listing 4.20 shows the basic logic.

```
public void AddTask(string task) {
    if (!activeTasks.Contains(task)) {
        activeTasks.Add(task);
        UpdateTaskUI(); // refresh UI
    }
}
public void UpdateTask(string updatedTask) {
    for (int i = 0; i < activeTasks.Count; i++) {
        if (activeTasks[i].Contains("Destroy")) {
            activeTasks[i] = updatedTask;
            UpdateTaskUI(); // refresh UI
            break;
        }
    }
}
public void CompleteTask(string completedTask) {
    if (activeTasks.Contains(completedTask)) {
        activeTasks.Remove(completedTask);
        UpdateTaskUI(); // refresh UI
    }
}
private void UpdateTaskUI() {
    if (activeTasks.Count > 0) {
        currentMissionText.text = "Current Mission:\n";
        foreach (string task in activeTasks) {
            currentMissionText.text += "- " + task + "\n";
        }
    } else {
        currentMissionText.text = "Current Mission:\nAll tasks completed!
    ";
    }
}

```

Listing 4.20: Mission List Logic

The core function of the introduction UI is to introduce the current celestial body to the player and show related knowledge, the task objectives, and its background. After the introduction, players can continue the game by clicking the "Continue" button, which

leads them into the game and begins the task. This method allows the game to convey solar system knowledge, also ensures that players are clear about the upcoming task goals and game progression (see Figure 4.6).

It is implemented by Unity `TextMeshPro` and `Button` components with scroll bar viewing. Title and Main Text are used to display the task title and main content. Using a scroll bar to view. A continue button is used to continue the game when it is clicked.

Figure 4.6: Introduction UI

4.4 Artificial Intelligent

Artificial intelligence is also a crucial component of the game, its primary goal is to simulate intelligent behaviour, making the game environment more realistic and challenging. Game AI is not only used to create decision-making processes for enemy objects, but also to adjust the game difficulty and provide a different player experience. This involves programming non-player characters (NPCs) to exhibit intelligent behaviours and make decisions [64].

In this game, AI is primarily used for the enemy's behaviour logic, including movement, target tracking, obstacle avoidance, and automatic attacks. For enemy movement, AI algorithms often use techniques like pathfinding to determine the optimal route in the game environment, considering terrain and obstacles. Techniques such as Finite State Machines (FSMs) can also be used to define different behaviour states for enemies, like patrolling, attacking, or retreating, these states can be triggered by game events.

The enemy's movement logic includes several parts: movement, target tracking, and obstacle avoidance. Through these functions, an enemy entity could autonomously move and actively seek targets.

An enemy entity could autonomously move and actively seek targets with dynamic obstacle avoidance mechanism. Target tracking includes the AI's ability to identify and follow the player or other targets within the game world [65]. Obstacle avoidance ensures that enemies can navigate the game environment without unexpected collisions. In the game, this is implemented by ray casting and label recognition. Ray casting is used for detecting obstacles in the enemy's path and generating corrections to deviate from them. For label recognition, where every object in the game environment has a different tag for identification, so AI will recognize if the object it detected is an obstacle or target.

Enemies have different objectives depending on the mission stage. For example, when fighting the player, the enemies' target is the player. When protecting the space cruiser, the enemies' target is the cruiser. When a target is detected, the enemy will move toward the target.

```
void MoveTowardsTarget(Vector3 targetPosition, float moveSpeed) {
    Vector3 directionToTarget = (targetPosition - transform.position).
    normalized;
    Vector3 avoidanceVector = Vector3.zero;
    float rayLength = obstacleAvoidanceRange;
    Vector3[] rayDirections = {transform.forward,
        (transform.forward + transform.right).normalized,
        (transform.forward - transform.right).normalized,
        (transform.forward + transform.up).normalized,
        (transform.forward - transform.up).normalized};
    // raycasting and avoidance
    foreach (Vector3 dir in rayDirections) {
        RaycastHit hit;
        if (Physics.Raycast(transform.position, dir, out hit, rayLength))
     {
            if (hit.transform != cruiser)
                avoidanceVector += (transform.position - hit.point).
    normalized;
        }
    }
    if (avoidanceVector != Vector3.zero){
        avoidanceVector = avoidanceVector.normalized * avoidanceStrength;
    }
    Vector3 finalDirection = directionToTarget + avoidanceVector;
    finalDirection = finalDirection.normalized;
    transform.forward = Vector3.Slerp(transform.forward, finalDirection,
    5f * Time.deltaTime);
    transform.position += transform.forward * moveSpeed * Time.deltaTime;
}

```

Listing 4.21: Enemy Movement

The enemy dynamic obstacle avoidance mechanism is implemented by directional raycasting and label recognition. Every object in the game environment has a different tag. When raycasting detects that an object in the movement direction is an obstacle,

and when the enemy is close enough to an object, there will be a correction direction is generated to deviate from the obstacle. It allows the enemy to smoothly bypass obstacles while keeping a tracking direction toward the target. This approach avoids the simple "wall collision" behavior and prevents unnatural pathing caused by excessive avoidance by setting the obstacle avoidance strength. Listing 4.21 shows the enemy movement logic.

The enemy attack feature is designed to allow enemies to engage players effectively during combat. This feature has a cooldown time and damage calculation. Listing 4.22 shows the enemy attack logic. The enemy will attack periodically with random damage values (10-20) and random attack intervals (2-5 seconds) to increase unpredictability (see Listing 4.23).

```
private IEnumerator FireLaserRoutine() {
    while (true) {
        float waitTime = Random.Range(minFireInterval, maxFireInterval);
        yield return new WaitForSeconds(waitTime);
        FireLaser();
    }
}
private void FireLaser() {
    if (laserPrefab != null && firePoint != null && player != null)
        Instantiate(laserPrefab, firePoint.position, firePoint.rotation);
}

```

Listing 4.22: Enemy Attack

```
if (collision.gameObject.CompareTag("LaserEnemy")) {
    HP -= Random.Range(10, 20);
    collisionAudioSource.PlayOneShot(isShotSound);
    TriggerDamageOverlay(); // Trigger red effect
}

```

Listing 4.23: Damage Value

4.5 Effects

The special effects component could enhance the game's immersion and interactivity through a series of visual and sound effect elements. Special effects not only improve the game's artistic presentation but also can effectively show the dynamic changes of game objects and events. For example, the brilliance of the sun is realized by particle systems, lighting effects, blur effects, and other techniques. Additionally, the combination of sound effects, such as the audio of interstellar flight, further enhances the atmospheric feel of the game. Through the application of these special effects, players can enjoy visually stunning moments while experiencing a more realistic gaming experience.

4.5.1 Visual Effects

Visual effects (VFX) are important for enhancing player immersion and the gaming experience. This game implements various visual effects, including damage feedback, explosion effects, and the camera's field of view with effects to simulate teleport.

The damage feedback is implemented by creating a UI image as the background for visual feedback to implement a masking effect, and combining it with hit audio (see Figure 4.7). Set the mask opacity to 10%, making it visible but not completely blocking the view. Also, configure the delay and fade-out times to align with the characteristics of human visual persistence (see Listing 4.24).

```
void TriggerDamageOverlay()
{
    StopAllCoroutines(); // avoid overlapping
    StartCoroutine(FadeDamageOverlay());
}
IEnumerator FadeDamageOverlay()
{
    overlayAlpha = maxAlpha; // display immediately
    UpdateOverlayAlpha();
    yield return new WaitForSeconds(0.1f); // keep displaying
    float elapsedTime = 0f;
    while(elapsedTime < fadeDuration) {
        overlayAlpha = Mathf.Lerp(maxAlpha, 0f, elapsedTime/fadeDuration)
    ;
        UpdateOverlayAlpha();
        elapsedTime += Time.deltaTime;
        yield return null;
    }
    overlayAlpha = 0f;
    UpdateOverlayAlpha();
}

```

Listing 4.24: Damaged VFX Code

For destroying effects, there are explosion effects. First, store several effects in a preset list. Then, when an object is destroyed, randomly select an explosion effect from the preset list to play with random rotation (see Figure 4.8 and Listing 4.25).

Figure 4.7: Damaged VFX

```
if(explosionEffects != null && explosionEffects.Count > 0) {
    int randomEffectIndex = Random.Range(0, explosionEffects.Count);
    ParticleSystem explosion = Instantiate(explosionEffects[
    randomEffectIndex], transform.position, Quaternion.identity
);
Destroy(explosion.gameObject, explosion.main.duration + explosion.main.
    startLifetime.constant);
}

```

Listing 4.25: Explosion VFX Code

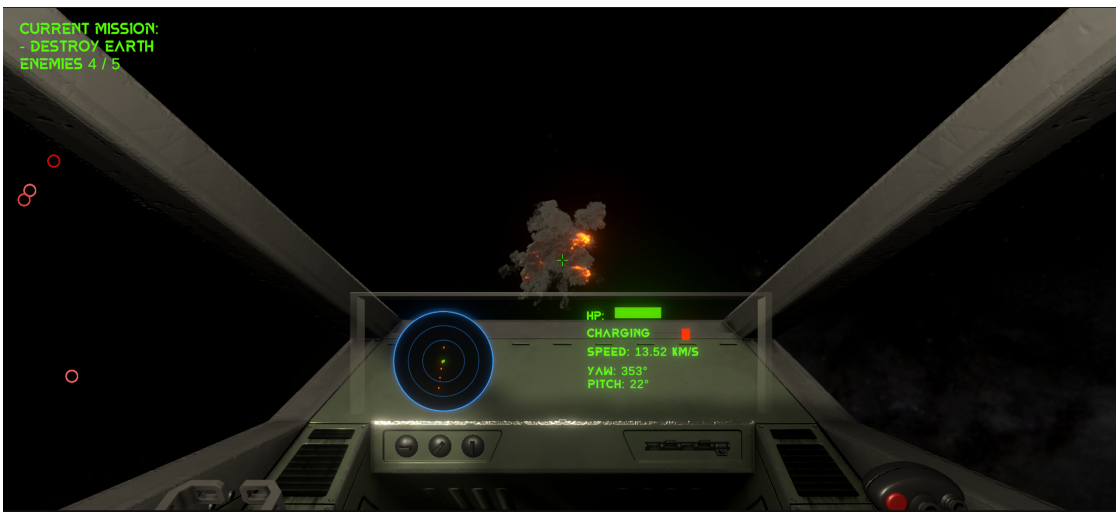

Figure 4.8: Explosion VFX

The teleportation complete sensory experience is combined with particle effects, camera field of view (FOV) changes (see Figure 4.9), and sound effects, making the teleportation process feel more vivid [66]. The teleportation code is shown in Listing 4.7.

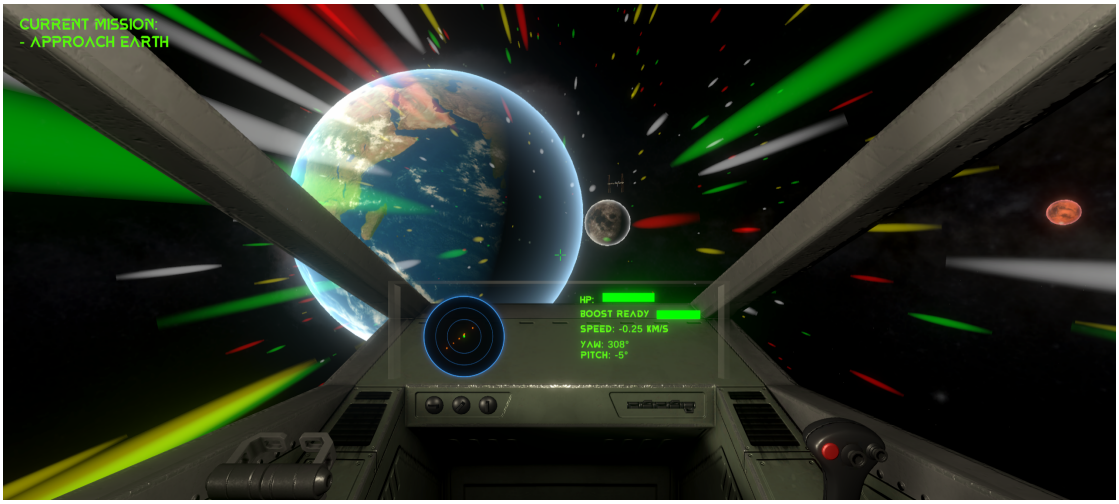

Figure 4.9: Teleport VFX

4.5.2 Sounds

The sound effect implementation is completed by `AudioSource` and `AudioClip`, the goal is to provide players an immersive gaming experience and enhance the atmosphere and interactivity for audio [67]. The sound implementation includes background music, task sound effects, and fighting sound effects. All sound effects are dynamically adjustable to adapt to different game scenes and task progress.

Various effects can be used to create the corresponding atmosphere. For example, fade-in and fade-out effects can simulate engine start-up and shut-down, and adjusting the volume can highlight the main audio. Listing 4.26 shows the sound fade-in and fade-out logic.

The sound component is mainly divided into several parts: Background music (BGM) is dynamically switched based on different task stages. There is a different BGM for different mission stage. Combat Audio, including shooting, and damage feedback. Engine audio, including acceleration, reverse thrust, and super boost. Effect audio, including teleport, task updates and collisions.

```
public IEnumerator FadeInAudio(AudioSource source, float targetVolume,
    float duration){
    source.volume = 0;
    while (source.volume < targetVolume){
        source.volume += Time.deltaTime;
        yield return null;
    }
}
public IEnumerator FadeOutAudio(AudioSource source, float targetVolume,
    float duration){
    float startVolume = source.volume;
    float timeElapsed = 0f;
    while (timeElapsed < duration){
        source.volume = Mathf.Lerp(startVolume, targetVolume, timeElapsed
    / duration);
        timeElapsed += Time.deltaTime;
        if (!source.isPlaying)
            yield break; // source stoped, break out
        yield return null;
    }
    source.volume = targetVolume;
    source.Stop();
}

```

Listing 4.26: Sound Fade-in and Fade-out

4.6 Simulation Implementation

This chapter introduces the scientific simulation implementation used to create the solar system environment in the game. The game has a huge and complex system, the solar system involves various aspects such as celestial body modelling, orbital mechanics, physics laws, and lighting rendering. To get a balance between education and entertainment, this project uses the Unity engine to develop a highly realistic and high-performance solar system simulation framework. By combining real astronomical data and modern graphics technology, players will get a realistic scientific learning experience. Figure 4.10 shows the project overview.

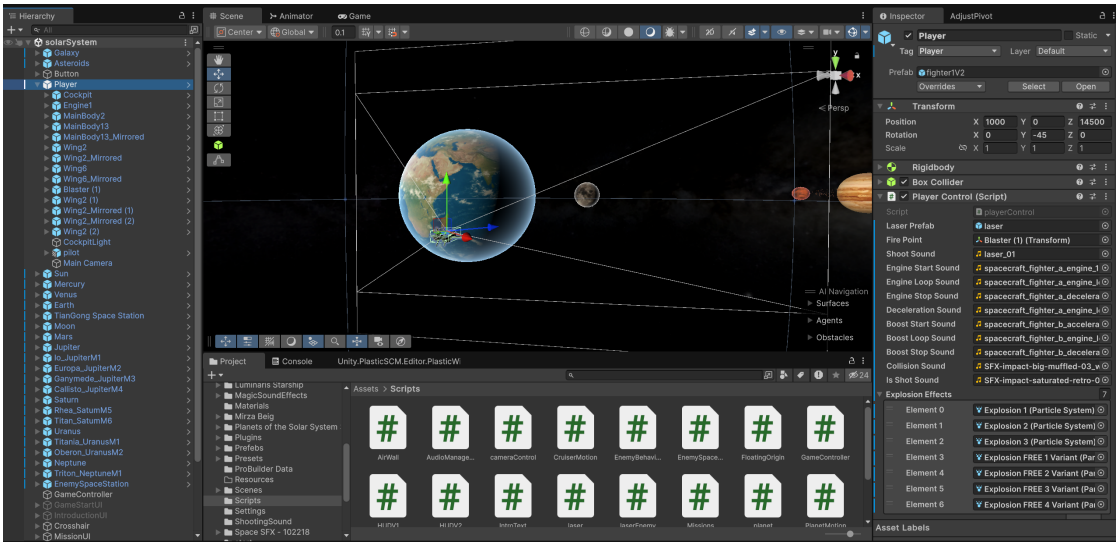

Figure 4.10: Project Overview

4.6.1 Solar System Environmental Simulation

The simulation of the solar system environment is built on the Unity engine with C# scripts, and with a scale ratio of 1:10,000 km for scene construction, which means that one unit in the game represents 10,000 kilometres in the real solar system. This scale ensures the spatial relationships preservation between celestial bodies and visual realism in this virtual universe. This section will introduce the specific implementation used for modeling, orbital motion, and visualization.

Celestial body modeling is the basis of the solar system celestial bodies, ensuring that players can visually observe the shape, size, and surface features of the celestial bodies in an intuitive way. The construction of the foundational structure serves as the starting point for celestial body modeling, determining the geometric form, relative scale, and visual representation of each planet and moon (satellite) in 3D space.

In this project, the geometric bases of celestial bodies are spheres based. They apply different mesh details and material layers to implement the celestial bodies visual representations. Each celestial bodies maintains an independent `GameObject` layer, which can facilitate subsequent physics property binding, script coding, and visual optimization.

There are main celestial bodies in Figure 4.11, including the sun, planets above, and moons (satellites) below. As mentioned in Chapter Three, there are ten moons created in this game. Additionally, there are two artificial satellites: As the TianGong will be the only human space station years later, it is designed as the location where the human base and the exit of the game are. After completing all missions, the player could go back to TianGong to exit the game.The enemy space station is designed as the enemy base in the solar system, and there is a simple exploration mission for this. They are shown in Figure 4.12.

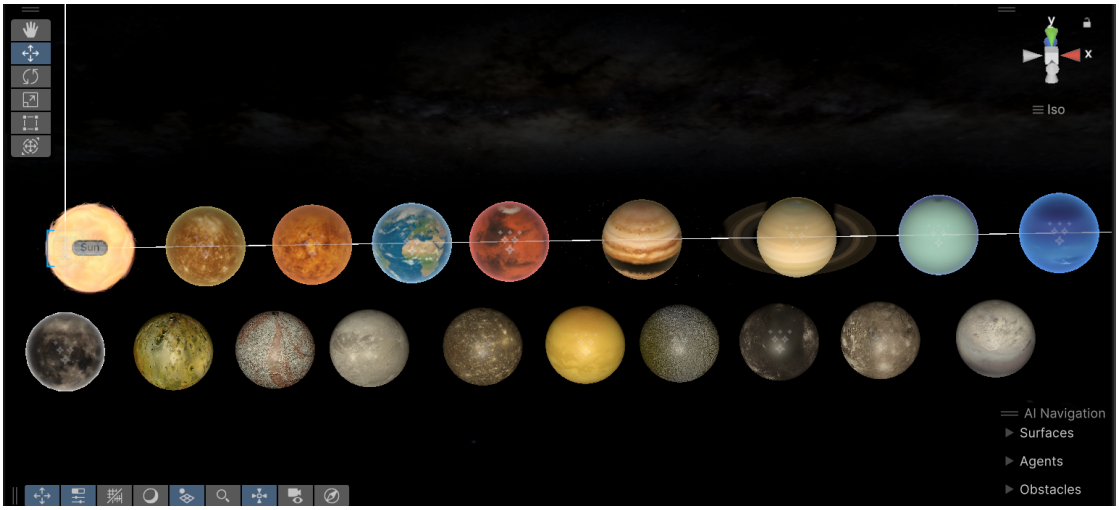

Figure 4.11: Main Celestial Bodies Models

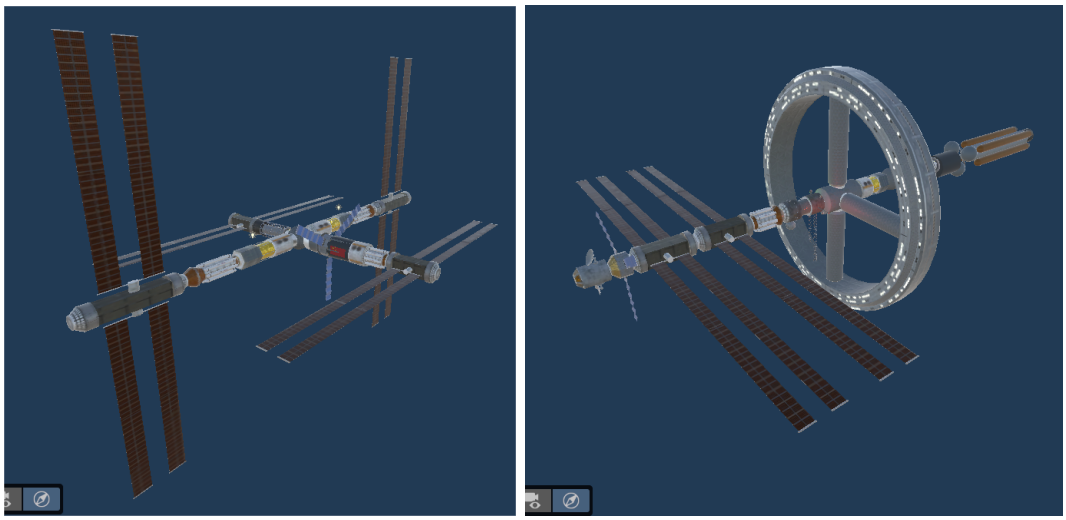

Figure 4.12: Tian Gong Space Station (left) and Enemy Space Station (right)

The High-Resolution texture mapping is created by using publicly available images from NASA, textures are processed with 4K or 8K resolution, including:

- **Albedo Map** used for base color texture.
- **Normal Map** for enhancing surface detail
- **Specular Map or Roughness Map** for reflective properties

There are more details to enhance the game's visual realism and educational value, simulating the environmental details with the following example: a dynamic cloud layer (transparent effect) is added to Earth and Mars (see Figure 4.13). Create the asteroid objects belt near Jupiter, which is implemented by a set of asteroid objects (see Figure 4.14). The Saturn's rings are implemented using multiple ring-shaped planes with a transparent texture (see Figure 4.15).

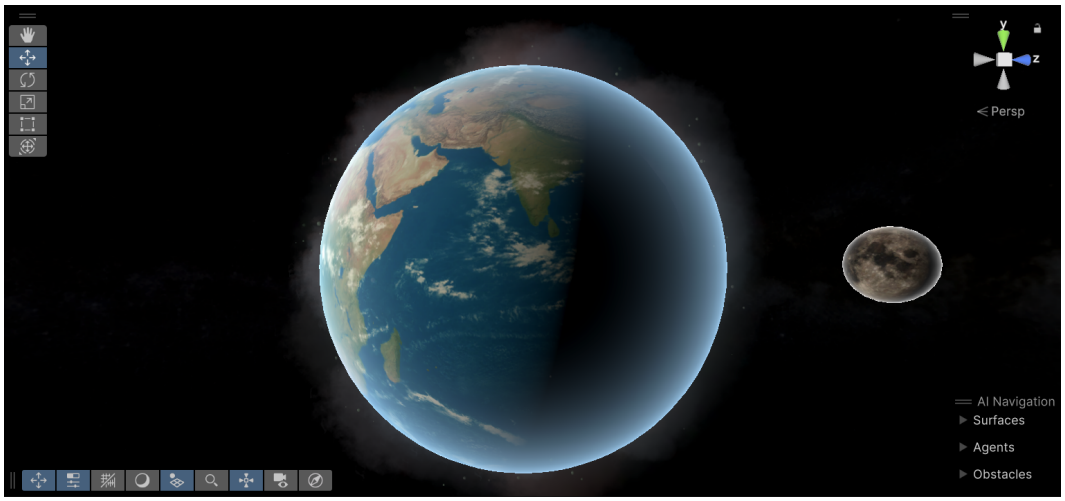

Figure 4.13: Earth Cloud Layer

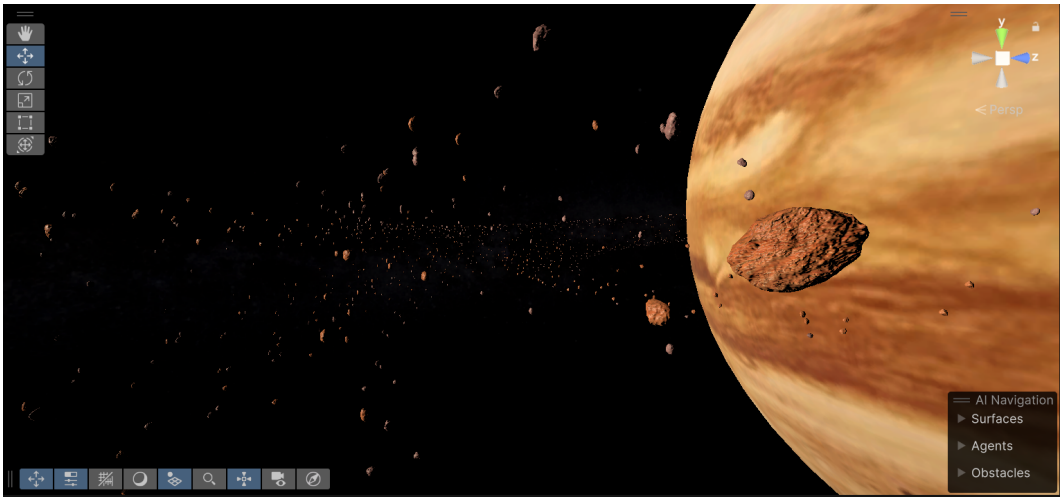

Figure 4.14: Asteroid Belt

Orbital motion is the key part to simulate the dynamics of the solar system, it will ensure the celestial bodies move along accurate trajectories based on Kepler's laws and Newton's Law.

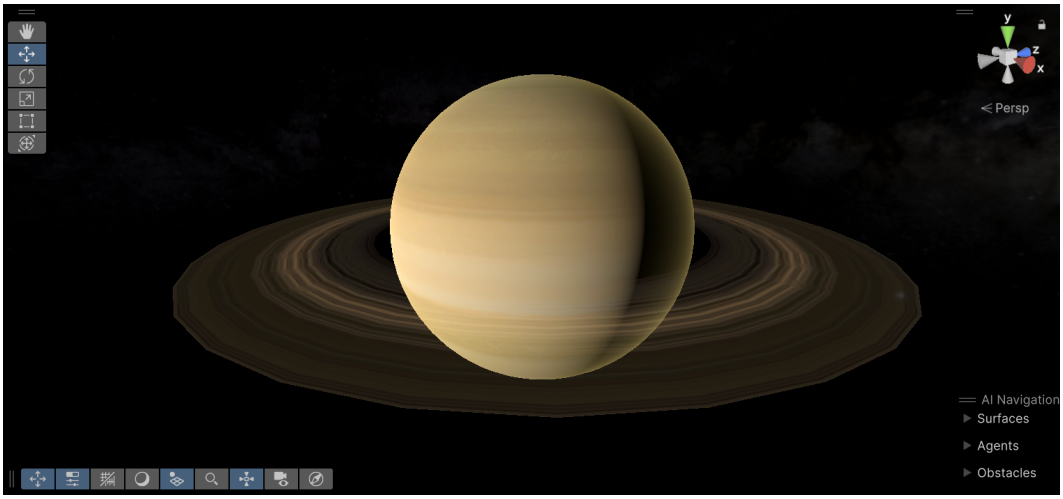

Figure 4.15: Saturn Ring

The motion of celestial bodies in the game is based on classical mechanics theory, with trajectory reconstruction incorporating NASA data. The celestial bodies' motions are based on Kepler's Laws and Newton's law.

Kepler's laws describe the motion of planets around the Sun. There are three laws of Kepler's Law: First Law describes that planetary orbits around the Sun are elliptical, with the Sun at one focus. Second Law describes that a planet moves slower when farther from the Sun and faster when closer. The third Law describes that the square of a planet's orbital period is proportional to the cube of its semi-major axis [21].

Newton's Law describes the motion of an object under the action of a force. It is shown as the following formula:

$$F = G\frac{m_1 m_2}{r^2}$$

Here, F is the gravitational force, G is the gravitational constant, m_1 and m_2 are the masses of two celestial bodies, r is the distance between them [68].

Advanced graphics technologies simulate lighting, atmospheric effects, and material properties to enhance players' experience and learning interest. Main implementations include Physically Based Rendering (PBR) and Dynamic Lighting.

Physically Based Rendering (PBR) is based on a cosmic skybox in Unity using the Universal Render Pipeline (URP), rendering materials based on realistic lighting models. Enhancing realism by using physical maps to implement surface textures, reflectivity, and roughness. Figure 4.16 and Figure 4.17 show the game scene and the in-game rendering of the sun.

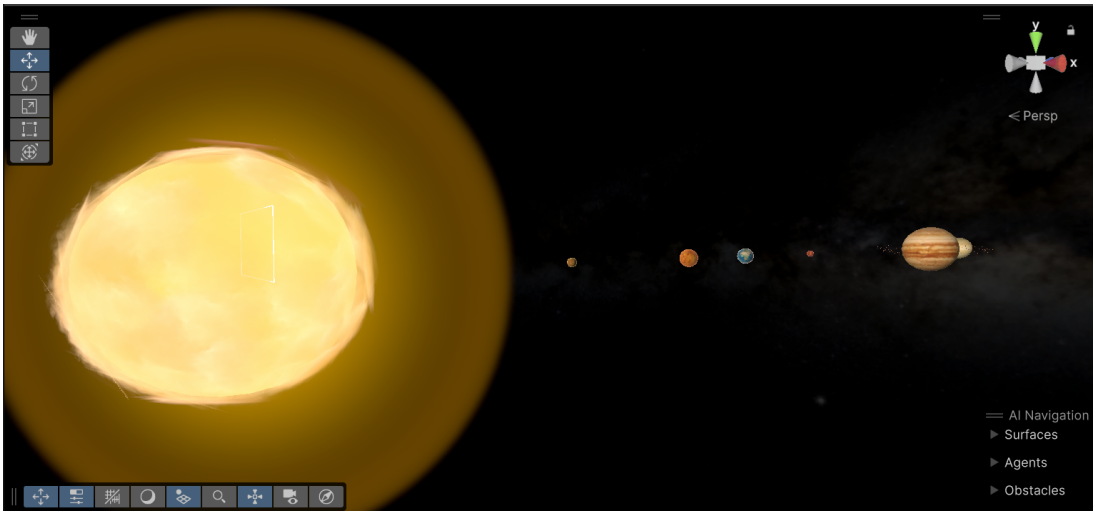

Figure 4.16: Game Scene Overview

Figure 4.17: Sun

4.6.2 Celestial Bodies Motion Simulation

This section explains how planets and moons (satellites) rotate and revolve following physical laws within the game. It also explains how configurable parameters enable individualized behavior for each celestial body. All celestial bodies' movements are defined and controlled by C# scripts, and they are adjusted based on the physical characteristics of each celestial body. This will make sure that their parameters align with real astronomical features.

The main motion parameters include several parts, for example, the basic parameters like size, and the motion parameters like rotation speed, there are the parameters:

- **Scale** decides the size of the celestial bodies models.
- **Distance From the Sun** decides the positions of celestial bodies, displayed by coordinates.
- **Orbit Target (Sun or Planet)** defines the target around which a celestial body revolves.
- **Orbit Speed (unit/sec)** controls the revolution speed
- **Rotation Speed (degree/sec)** controls the rotation speed
- **Axial Tilt (degree)** defines the axial tilt of rotation
- **Orbit Inclination (degree)** defines the inclination of the orbital plane

For example, Listing 4.27 shows the Earth's parameters.

```
rotationSpeed = 0.004167f;
orbitSpeed = 0.002978f;
axialTilt = 23.5f;
orbitInclination = 0.0f;
target = Sun.transform;

```

Listing 4.27: Earth Parameters

Different celestial bodies have different initialization logics. For the sun and planets, their positions are set in the game world manually, and the moons' (satellites) positions are initiated by script. The sun is always at the world's center (0, 0, 0). Celestial bodies' initial positions are set related to the Sun using the game's scale. For instance, Earth's real distance (149,597,870 km) becomes about 14,960 units in Unity, placing it at (0, 0, 14960). All planets and the Sun will be on the same line initially, but the moons' (satellites) initial positions are calculated and generated by the script based on their parameters, like orbital radius and inclination. This could avoid overlap or the same spawn points (see Listing 4.28).

```
if (CompareTag("Satellites"))
{
    transform.rotation = Quaternion.Euler(axialTilt, 0, 0);
    initialOrbitAngle = Random.Range(0f, 360f);
    Vector3 orbitPlaneNormal = Quaternion.Euler(0, 0, orbitInclination) *
     Vector3.up;
    Vector3 orbitDirection = Quaternion.AngleAxis(initialOrbitAngle,
    orbitPlaneNormal) * (Quaternion.Euler(0, 0, orbitInclination) *
    Vector3.right);
    transform.position = target.position + orbitDirection * orbitRadius;
}

```

Listing 4.28: Celestial Bodies Init Logic

`Quaternion.Euler(axialTilt, 0, 0)` controls the initial orientation of the celestial body's rotation axis. If the object is a satellite, generate its position along the orbit, and the orbital angles are randomly distributed by `Random.Range` to avoid overlap. The special parameter is set for satellite `orbitRadius`, which is the distance from the satellite to its planet. The spatial computation of orbital inclination is completed by quaternion rotation, which can ensure the establishment of non-coplanar orbits in a 3D space. For example, Listing 4.29 shows the moon's parameters.

```
rotationSpeed = 0.012f;
orbitSpeed = 0.0001022f;
axialTilt = 1.54f;
orbitInclination = 5.145f;
orbitRadius = 734f;
target = Earth. transform;

```

Listing 4.29: Moon parameters

As mentioned above, the motion of celestial bodies follows Kepler's laws. In the real world, celestial bodies will run around a central body (such as the Sun) along the elliptical paths. However, in a game environment, the world scale will be much smaller than the real universe, and the effect of orbital eccentricity will become negligible. Therefore, these orbits are abstracted as a circular path to reduce computational complexity, and it can still preserve the core physical characteristics such as orbital inclination, orbital period. In this section, the simulation of celestial motion based on Kepler's laws in the game will be explained.

For building a physically realistic orbital system in a game, it is necessary to construct a rotation axis with inclination, which is used for the orbital normal direction for the revolution. Firstly, initiate the inclined axis, and each celestial body will revolve around this axis, this will result in an inclined orbital motion (see Listing 4.30). In each frame, the revolution angle is updated based on the defined orbit speed (see Listing 4.31).

```
Vector3 inclinedAxis = Quaternion.Euler(0, 0, orbitInclination) * Vector3
    .up;

```

Listing 4.30: Init Inclined Axis

```
transform.RotateAround(target.position, inclinedAxis, orbitSpeed *
    timeScale * Time.deltaTime);

```

Listing 4.31: Update Revolution

The rotation simulation is based on the configured `rotationSpeed`, and the object rotates around `Vector3.up` each frame. The revolution is implemented by Unity's `RotateAround`, the target as the center, and a tilted `Vector3.up` (after applying orbital inclination) as the axis, then achieving inclined orbital motion. The inclined orbital axis is constructed using `Quaternion.Euler(0, 0, orbitInclination)`, allowing for a layered distribution of orbits with different inclinations (see Listing 4.32).

```
// Rotation: Rotation around its own Y axis
transform.Rotate(Vector3.up, rotationSpeed * timeScale * Time.deltaTime);
// Calculate the inclined axis
Vector3 inclinedAxis = Quaternion.Euler(0, 0, orbitInclination) * Vector3
    .up;
// Revolution: Rotation around an inclined axis
transform.RotateAround(target.position, inclinedAxis, orbitSpeed *
    timeScale * Time.deltaTime);

```

Listing 4.32: Motion Running

4.6.3 Flight Simulation

Spaceflight simulation is a key component of this project that combines scientific accuracy and interactive gameplay. This section introduces how the fighter moves in space based on real-world physical laws.

Players will use fundamental principles of dynamics to complete missions by piloting the fighter. The fighter's movement also follows Newton's laws with assist functions. The movement of the fighter in space follows real physical principles, including inertial flight, and propulsion system operation.

Without air resistance, the fighter maintains its velocity in the absence of thrust. If no external forces (like boosting) act on it, the fighter will drift in space. If an impact is received, the impacting force will always be valid until the player uses the stabilization function to counteract it and restore balance. As mentioned in Chapter 4, players can

stabilize the fighter by `KeyCode.R`, this function is designed for gameplay balance, and it simulates an effect similar to that of a reaction wheel. The code could be seen in Listing 4.5.

The vector engine propulsion is also simulated in this game. Acceleration is simulated through direct control of translational force in the forward direction [62][63]. When the engine is activated (via `KeyCode.E`), acceleration is calculated as Listing 4.33 showed. This reflects a throttle-based increase in speed, limited by `maxSpeed`, simulating realistic propulsion constraints, which means closer to get the maximum speed, the acceleration becomes slower. In a vacuum, deceleration is controlled manually with a constant rate to simulate a reverse thruster by pressing `KeyCode.S`. The code could be seen in Listing 4.3.

```
1 float acceleration = baseAccelerationRate * (1 - (movementSpeed /
      maxSpeed));
2 movementSpeed += acceleration * Time.deltaTime;
3
```

Listing 4.33: Acceleration Simulation

As Chapter 4 mentioned, mouse input is used to simulate joystick controls for the fighter's attitude, and the FOV effects are used to simulate acceleration and deceleration [66]. The fighter's yaw and pitch are controlled by the mouse. The inputs will be translated into Quaternion rotations for each frame, with real-time response for Mouse X and Mouse Y, to simulate a realistic pitch/yaw feel. Players use mouse inversion to simulate realistic joystick pitch control (see Listing 4.34). Additionally, the field of view (FOV) dynamically changes based on motion state (acceleration, deceleration, boost), simulating a visual perception of speed. When the fighter accelerates, the FOV becomes larger, when the fighter decelerates, the FOV becomes smaller [66]. The effect is illustrated in Figure 4.9.

```
1 Quaternion yawRotation = Quaternion.AngleAxis(yawInput, Vector3.up);
2 Quaternion pitchRotation = Quaternion.AngleAxis(pitchInput, Vector3.right
      );
3 transform.rotation *= yawRotation * pitchRotation;
4
```

Listing 4.34: Attitude Control Simulation

5 Evaluation

This chapter summarizes the practical results and insights obtained during the development and internal testing of the game. It focuses on two key aspects: performance optimization and gameplay testing feedback. Through iterative testing and debugging, several technical challenges were identified, particularly related to rendering, coordinate systems, collision, and physics simulation. Related optimizations were implemented to ensure smooth gameplay. The gaming testing feedback is collected by a questionnaire.

5.1 Performance Optimization

Based on the ongoing development and testing phases, several performance-related issues have been identified that require optimization to ensure a stable and smooth gaming experience.

In large world games, floating-point precision errors accumulate as the player moves farther from the world origin point (0,0,0), resulting in UI jittering, inaccurate physics. To fix this, a Floating Origin System was implemented to dynamically recenter the coordinate space around the player (see Listing 5.1) [69].

```
if (distanceFromOrigin > thresholdDistance){
    ShiftWorld(playerPosition);
}

// in ShiftWorld() method
{
    // get all activated objects
    GameObject[] allObjects = FindObjectsOfType<GameObject>();
    // move each object
    foreach (GameObject obj in allObjects)    {
        // skip player and cam
        if (obj == gameObject || obj == playerCamera.gameObject)
            continue;
        // ensure only move root object, avoid sub-object moving repeatly
        if (obj.transform.parent == null)        {
            obj.transform.position -= shiftAmount;
        }
    }
}

```

Listing 5.1: Floating Origin Point

During each frame update, the distance between the player and the world origin is calculated. When this distance exceeds a predefined threshold (10,000 units in this implementation), the system executes a world shift, moving all relevant objects, including physics bodies, by the same offset in the opposite direction of the player's current position. This effectively returns the player to the origin without altering their relative position within the game world. This method allows rendering a big world game environment without unstable. It also decouples visual and physics precision from absolute world coordinates, making it especially suitable for big world environments like the solar system.

Dynamic Level of Detail (LOD) rendering is used for optimizing performance by dynamically adjusting the geometric complexity to ensure realistic visuals of celestial bodies based on the player's camera distance. When a celestial object is far from the player, the player can only see a simplified version of its geometry, such as a simple sphere with a low-quality texture. As the player gets closer, higher resolution models and textures are rendered [70]. Figure 5.1 shows the comparison between low LOD and high LOD. In the solar system simulation, this method allows the efficient and stable rendering of planets and moons, not only ensuring a smooth experience but also providing details in closer observations.

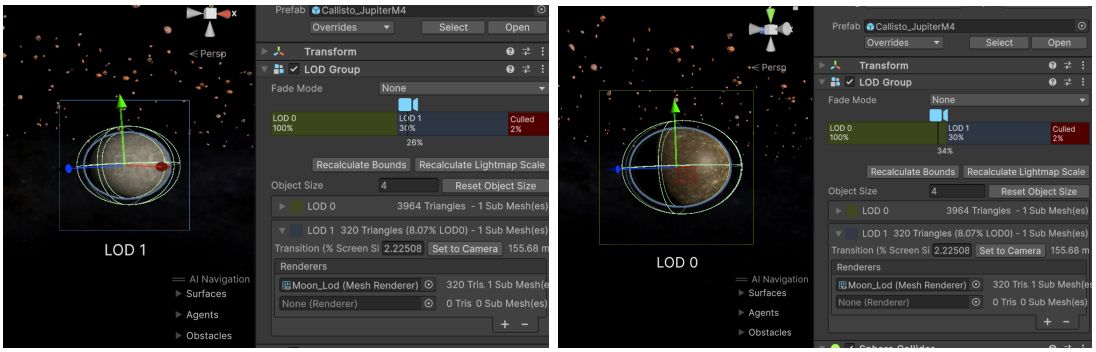

Figure 5.1: Low LOD (left) and High LOD (right) Comparison

To define the specific boundaries and zones in space, such as planets. An airwall system is built by Unity's physics engine based on sphere collision. The airwall is implemented by a spherical collision trigger, which detects if the fighter reaches the defined boundary. The airwall size is set as 1.2 times the celestial bodies. When the fighter's collider reaches the airwall's `SphereCollider`, the system calculates the direction vector from the center of the airwall to the player's position. The player's position is then reset to the surface of the airwall sphere in that direction. This method effectively prevents crashing between the fighter and celestial bodies (see Figure 5.2 and Listing 5.2).

```
1 private void OnTriggerEnter(Collider other){
2     if (other.CompareTag("Player")){
3         Vector3 direction = other.transform.position - transform.position
      ;
4         other.transform.position = transform.position + direction.
      normalized * GetComponent<SphereCollider>().radius;
5     }
6 }
7
```

Listing 5.2: Airwall Logic

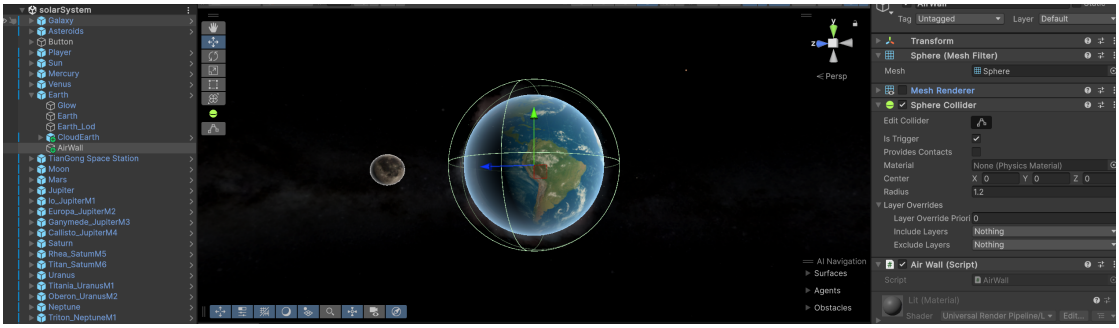

Figure 5.2: Airwall Setting

In the early design phase, a gravity simulation was considered for celestial bodies in the game world. However, due to the huge scale difference between the game world and the real solar system, and considering gameplay balance and technical complexity, the gravity system was removed from the final implementation.

Gravity calculations for multiple celestial bodies would increase hardware usage, particularly for N-body interactions or long-distance trajectory prediction. Removing gravity reduces computational overhead and simplifies the physics system. Realistic gravity introduces complex orbital mechanics with physically accurate, compared to the real solar system, the game world operates on a much smaller spatial scale. In such an environment, the gravitational influence of celestial bodies becomes insignificant unless artificially amplified. If players want to observe the effects of gravity obviously, it is necessary to artificially amplify the parameters. However, in this case, it will influence the balance of the game environment. Artificially editing the gravity parameters will result in unstable fighter motion. In this case, it may cause a worse game experience for players. By removing gravity, the fighter's movement becomes fully controlled by the player, improving responsiveness and maintaining a stable pilot experience.

After these optimizations, there is another UI system based on the basic game scene with a more assistant system, including teleportation hints and an advanced radar that will display the nearest planet, so that players would not get lost (see Figure 5.3). This

system provides more assistant information, so that make this game is easier to start with a smoother learning curve.

Figure 5.3: Advanced Cockpit

5.2 Game Testing and Evaluation

The game effectiveness and player experience evaluation is tested by user questionnaire testing and finally collected 13 responses. The participants were aged between 20 and 30 years old. Most of them are students, and a smaller number of them are working professionals. Additionally, there are 11 male and 2 female participants. All participants had fundamental gaming experience, but the male players have much more gaming experience.

There is a questionnaire survey to collect player feedback about various aspects of the game evaluation. There are 10 questions totally in this questionnaire, and they cover several areas about this game, such as core game mechanics, audiovisual experience, challenge level, controls, mission system, educational value, integration of learning and gameplay, the balance between learning and gaming, and if the game logic is reasonable and understandable. Each question used a Likert Scale style with four options: “Strongly Disagree”, “Disagree”, “Agree”, and “Strongly Agree”.

The results of the questionnaire show that players have an overall positive attitude of the game. For example, the questions 1, 3, 4, and 5 received 10 positive responses, the questions 7 and 9 received 11 positive responses, the questions 2 received 12 positive

responses, and the questions 6, 8, and 10 received 13 positive responses (total positive feedback). The one-sided binomial test determines if the positive feedback was statistically significant for each question. The test calculates the probability of getting the positive responses numbers (Agree and Strongly Agree), if players were choosing randomly between positive and non-positive feedback, then get the null hypothesis: p=0.5. The test function is the `binomtest()` from Python, Listing 5.3 showed the whole function.

```
1 binomtest(positiveResponsesNumber, totalResponsesNumber, p=0.5,
      lternative='greater')
2
```

Listing 5.3: p-value calculation function

If the result is statistically significant, the small p-value is obtained, which is less than 0.05. This will proof that the collected positive feedback for a question is not too possible occurred by randomly choosing, so players really had a positive attitude for the specific aspect of the game.

Table 5.1 summarizes the p-value results for each question. As the table showed, the positive feedback for all questions reached statistical significance ($p < 0.05$). This can strongly prove that the positive ratings were not random. The extremely low p-values for questions related to learning (Q6, Q8) and story (Q10) prove the success of learning and balancing parts.

Overall, the questionnaire and the statistical analysis display significantly positive feedback for the game. Players will get the core engaging gameplay and the immersive audiovisual experience. Importantly, the game met the educational goals effectively, players learned about the Solar System, the learning content was clear, and the integration with gameplay and learning reached the overall balance.

5.3 Discussion of the Results

The results of the internal testing and the subsequent player questionnaire provide valuable insights into the performance and quality of the developed game. The performance optimization efforts, particularly the implementation of the Floating Origin System and Dynamic Level of Detail (LOD) rendering, have successfully fixed the main potential technical problems in large-scale space simulations. The Floating Origin System function that maintains numerical and system stability is critical for a game set in a solar system, preventing issues like UI jittering and coordinate incorrect that will impact player experience. The use of LOD rendering ensures that the game remains performance while still offering visual detail when players are in closer proximity to celestial bodies, and achieves a necessary balance between graphical quality and computational efficiency. The airwall system, while a simplification of realistic collision, provides a functional and computationally inexpensive method to prevent players from colliding directly with planets, maintaining gameplay flow. The decision to ignore a full gravity simulation, as

discussed, was a choice to get the balance of realism, technical complexity, and player control, resulting in more predictable and stable fighter movement.

Q#	Question Summary	Positive Feedbacks	P-value	Significant
1	The combat mechanics make me feel excited and engaged.	10	0.0461	Yes
2	The visual effects and sound design enhance my immersion	12	0.0017	Yes
3	The game play is too challenging	10	0.0461	Yes
4	The controls are smooth and easy to learn	10	0.0461	Yes
5	The reward system motivates me to keep playing	10	0.0461	Yes
6	I learn knowledge about the Solar System	13	0.0001	Yes
7	The combination of the study and gameplay is done well	11	0.0112	Yes
8	The study part is clear and easy to read and learn	13	0.0001	Yes
9	There is a good balance between learning and playing	11	0.0112	Yes
10	The game story is logical and engaging	13	0.0001	Yes

Table 5.1: P-value result for each question

The player feedback questionnaire strongly supports the notion that these technical foundations translate into a positive player experience. The statistical analysis using the one-sided binomial test revealed that positive responses were statistically significant for all ten questions, indicating that the positive feedback is unlikely to be due to random chance. This overall positive feedback is a key point.

Specifically, the exceptionally low p-values for questions related to learning about the Solar System, like Q6 and Q8, and the game story, like Q10, are particularly encouraging. This strongly suggests that the educational goals of the project were effectively satisfied, and the storyline provided a reasonable logic and framework for the gameplay. The significant positive feedback on the combination of study and gameplay and the balance between learning and playing, which are Q7 and Q9, further shows the design of integrating educational content into the interactive gaming experience, rather than presenting it as an isolated element.

Besides the educational aspects, the statistically significant positive responses for questions concerning core gameplay mechanics, for example, Q1 is related to combat engagement, Q2 is related to audiovisual experience, Q3 is related to challenge level, Q4 is related to controls, and Q5 is related to the reward system. They indicate that fundamental aspects of the game design are highly regarded. This suggests that the game is not only effective as an educational tool but also succeeds in being an enjoyable and engaging game experience.

However, it is important to acknowledge the limitations of the current evaluation study. Firstly, the sample size of participants is relatively small, which is 13. The statistical significance tests provide confidence that the results are not random within this sample, but a larger participant group would be necessary to infer these findings to a wider potential players. Secondly, the participant demographics were limited to a specific age range (20-30), primarily students, and males. This demographic profile may not be representative of the wider range of players for the game, potentially introducing bias into the feedback. Different age groups, educational backgrounds, or gaming experience levels might have different opinions.

Furthermore, the evaluation depends on a structured questionnaire using a Likert scale. It is efficient for collecting data on specific aspects, but this method may lack the richness of feedback. Some open questions or interviews could have provided deeper insights, for example, why players felt a certain way, or areas for improvement that were not collected by the choice questions.

In conclusion, the evaluation results indicate that the technical optimizations implementations have created a stable platform for an engaging game with effectively integrates educational content about the Solar System. The positive feedback from various aspects, like learning and story, highlights key strengths. Nevertheless, future evaluations could focus on a larger and more diverse participant range and different statistical methods, so that get a more comprehensive understanding of player experience and refine the game design and educational impact.

6 Conclusion and Future Work

This thesis presented the design, development, and evaluation of this game ("Wentian") aimed at enhancing learning about the solar system through an engaging and interactive experience. Using the principles of Game-Based Learning (GBL), the project successfully integrated scientific accuracy with immersive gameplay to solve the challenges associated with teaching abstract astronomical concepts. The game provides an interactive environment where players explore planetary science and defend solar system resources through a series of missions.

The core contributions of this research are the creation of a realistic and scientifically simulated solar system environment within a game framework, coupled with a mission-driven gameplay approach designed to facilitate astronomical knowledge learning. Key aspects of educational game design include game mechanics, virtual environments, and scientific simulations. By utilizing real-world astronomical data, the game accurately represents celestial bodies, like planetary atmospheres, and orbital mechanics. A series of linear story missions embeds learning within the gameplay.

The development process successfully solved challenges in balancing scientific accuracy with user engagement. The implementation details covered the simulation of the solar system environment, including celestial body modeling, orbital motion based on Kepler's and Newton's laws, and flight simulation based on realistic physical principles like inertial flight and propulsion. Performance optimizations such as a Floating Origin System and Dynamic Level of Detail (LOD) rendering were crucial for ensuring a stable and smooth experience in a large-scale environment.

The evaluation is made through user testing with a questionnaire, which demonstrated a significantly positive reception of the game. Players found the core gameplay engaging, easy to control and understand, and the immersive audiovisuals. Crucially, the educational objectives were met effectively, participants indicating that they learned about the solar system, found the learning content clear, and were satisfied with the integration of learning and gameplay, as well as the overall balance and story. The statistically significant positive feedback about various aspects of the game validates its potential as a valuable tool for STEM education. The results support the potential of serious games to make complex scientific concepts more understandable and enjoyable, contributing to the digital education field.

This project had a successful development and positive evaluation, but several areas in educational impact and gaming experience still could be updated to enhance its quality. The game content could be expanded, it could incorporate more celestial bodies, such as additional dwarf planets, asteroids, and comets, to provide a more comprehensive representation of the solar system. Developing new mission types and challenges related to these objects would further enrich the learning experience.

The scientific simulation could be updated. There could be the simulation of gravitational interactions between celestial bodies without reducing performance obviously. This could involve optimized N-body simulation algorithms or simplified gravitational models that still convey the fundamental principles. Implementing more detailed atmospheric effects and geological features based on the latest scientific data will also enhance realism. There could be more extensive user studies involving a wider range of age groups, educational backgrounds, and learning environments to gather more diverse feedback and assess the game's effectiveness across different contexts. Comparing learning outcomes with traditional teaching methods through controlled experiments would provide stronger evidence of the game's impact, but this requires collecting the traditional teaching data.

Multiplayer and collaborative learning could be added. Introduce multiplayer features to allow students to explore the solar system together, collaborate on missions, and engage in competitive challenges. Collaborative gameplay could foster peer learning and communication skills. VR/AR exploration integration could be implemented. Virtual reality (VR) or augmented reality (AR) technologies will provide a more immersive and intuitive learning and gaming experience [17][18]. VR could allow for a more direct sense of scale and spatial relationships within the solar system. Advanced AI for dynamic challenges could be developed. AI with more complex enemy behaviors would create more dynamic and adaptive challenges. This could include more complex navigation, strategic attacking or retreating, and coordinated enemy movements [65]. Further enhancing the visual effects and sound design could increase player immersion and engagement [67]. This could involve more detailed environmental effects, advanced lighting and rendering techniques, and a more dynamic soundscape. Implement an adaptive learning system that adjusts the difficulty and content of missions based on the player's performance and demonstrated understanding of astronomical concepts. This could personalize the learning path for each user.

After these updates, "Wentian" can evolve into a more powerful educational tool with richer content, so that it can provide deeper insights into the solar system and inspire a greater interest in science and technology learning.

Bibliography

[1] G. Passante, S. J. Pollock, and H. R. Sadaghiani, "Student reasoning about measurement uncertainty in quantum versus classical physics labs," *Phys. Rev. Phys. Educ. Res.*, vol. 20, p. 010135, Mar 2024.

[2] U. Eriksson, "Disciplinary discernment: Reading the sky in astronomy education," *Phys. Rev. Phys. Educ. Res.*, vol. 15, 2019.

[3] Y. Liu, M. Wu, and J. Zhao, "Integrating virtual simulation technology into science education: A review of benefits, challenges, and future directions," *Asia Pacific Journal of Education*, pp. 1–17, 2024.

[4] R. Situmorang, H. Suwono, Munzil, H. Susanto, C.-Y. Chang, and S.-Y. Liu, "Learn biology using digital game-based learning: A systematic literature review," *Eurasia Journal of Mathematics, Science and Technology Education*, vol. 20, no. 6, p. em2459, 2024.

[5] P. Mikrouli, K. Tzafilkou, and N. Protogeros, "Applications and learning outcomes of game-based learning in education," *International Educational Review*, vol. 2, no. 1, pp. 25–54, 2024.

[6] R. E. Mayer, "Computer games in education," *Annual review of psychology*, vol. 70, no. 1, pp. 531–549, 2019.

[7] T. Anastasiadis, G. Lampropoulos, and K. Siakas, "Digital game-based learning and serious games in education," *International Journal of Advances in Scientific Research and Engineering*, vol. 4, no. 12, pp. 139–144, 2018.

[8] E. Team, "What is gbl (game-based learning)?," *EdTechReview*, March 2017.

[9] K. Squire, "Video games in education," *Int. J. Intell. Games & Simulation*, vol. 2, no. 1, pp. 49–62, 2003.

[10] C. Egert and A. Phelps, "Balancing entertainment and educational objectives in academic game creation," in *Proceedings of the 2022 International Conference on Game-Based Learning*, 2022.

[11] G. Cadiz, G. Lacre, R. Delamente, and T. Diquito, "Game-based learning approach in science education: A meta-analysis," *International Journal of Social Science and Human Research*, vol. 6, no. 3, pp. 1856–1865, 2023.

[12] Y. Qian, *3D Multi-User Virtual Environments in Science Education: Potential and Challenges*. IGI Global, 2015.

[13] G. Falloon, "Using avatars and virtual environments in learning: what do they offer?," *British Journal of Educational Technology*, vol. 41, no. 2, pp. 108–122, 2011.

[14] K. C. Yu, "Digital full-domes: The future of virtual astronomy education," 2005. Accessed via CiteSeerX.

[15] J. Keet, "Immersive learning in science with vr," 2023. Accessed May 15, 2025.

[16] H. Matovu, D. A. K. Ungu, M. Won, C.-C. Tsai, D. F. Treagust, M. Mocerino, and R. Tasker, "Immersive virtual reality for science learning: design, implementation, and evaluation," *Studies in Science Education*, vol. 59, no. 2, pp. 205–244, 2023.

[17] R. S. M. M. Magdalena Kersting, Jackie Bondell, "Virtual reality in astronomy education: Reflecting on design principles and learning outcomes," *Research in Learning Technology*, vol. 31, no. 1, pp. 1–15, 2023.

[18] M. N. Fabin Rasheed, Prasad Onkar, "Immersive virtual reality to enhance the spatial awareness of students," *Journal of Educational Psychology*, vol. 108, no. 5, pp. 728–739, 2016.

[19] K.-S. Grant Cooper, Li Ping Thong, "Transforming science education with virtual reality: an immersive representations model," *Educational Media International*, vol. 61, no. 1, pp. 1–18, 2024.

[20] A. Ghosh, "Challenges in educational game development," *International Journal of Darshan Institute on Engineering Research and Emerging Technology*, vol. 11, no. 1, pp. 54–60, 2022.

[21] J. L. Russell, "Kepler's laws of planetary motion: 1609–1666," *The British journal for the history of science*, vol. 2, no. 1, pp. 1–24, 1964.

[22] P. Musgrave, "N-body physics blog." `https://nbodyphysics.com/blog/`. Accessed: 2025-04-25.

[23] R. Haque, E. Aliya, I. Song, and N. Weliweriya Liyanage, "3d astronomy simulations for learning: Simulating solar and lunar eclipses," 04 2024.

[24] NASA/JPL Solar System Dynamics, "Solar system objects," Aug. 2023. Accessed: 2023-08-14.

[25] S. Hadden and M. J. Payne, "Terrestrial exoplanet simulator: an error optimal planetary system integrator," *Monthly Notices of the Royal Astronomical Society*, vol. 504, no. 1, pp. 678–693, 2021.

[26] B. E. Group, "What is task-based learning? a guide to the popular teaching method." `https://bridge.edu/tefl/blog/what-is-task-based-learning/`, 2020. Accessed: 2025-04-25.

[27] K. Susman and J. Pavlin, "Improvements in teachers' knowledge and understanding of basic astronomy concepts through didactic games," *ResearchGate*, 2020. Accessed: 2025-04-25.

[28] J. Pavlin and K. Susman, "Game-based learning to engage students with physics and astronomy using a board game." `https://www.researchgate.net/publication/330062584_Game-Based_Learning_to_Engage_Students_With_Physics_and_Astronomy_Using_a_Board_Game`, 2018. Accessed: 2025-04-25.

[29] K. Squire and H. Jenkins, "Harnessing the power of games in education," *Insight*, vol. 3, no. 1, pp. 5–33, 2003.

[30] A. Z. Abbasi, D. H. Ting, and H. Hlavacs, "Engagement in games: Developing an instrument to measure consumer videogame engagement and its validation," *International Journal of Computer Games Technology*, vol. 2017, no. 1, p. 7363925, 2017.

[31] H. Mouaheb, A. Fahli, M. Moussetad, and S. Eljamali, "The serious game: what educational benefits?," *Procedia-Social and Behavioral Sciences*, vol. 46, pp. 5502–5508, 2012.

[32] R. Rogers, "The motivational pull of video game feedback, rules, and social interaction: Another self-determination theory approach," *Computers in Human Behavior*, vol. 73, pp. 446–450, 2017.

[33] J. Sutton, "Mihaly csikszentmihalyi: The father of flow (2021)," 2021.

[34] Legends of Learning, "7 advantages of game-based learning strategies," 2021. Accessed: 2025-04-25.

[35] S. Szil á gyi, E. Palencs á r, A. Krei, and Z. Trk, "Examining the effectiveness of non-digital game-based learning among university computer science students on the topic of improper integrals," *Education Sciences*, vol. 15, no. 2, p. 132, 2025.

[36] P. Wouters, G. Camp, and J. van der Meijden, "Effects of games in stem education: a meta-analysis on the impact of game-based learning on students' academic achievements in primary and secondary education," *Journal of Curriculum Studies*, vol. 54, no. 4, pp. 1–23, 2022.

[37] S. Adipat, K. Laksana, K. Busayanon, A. Asawasowan, and B. Adipat, "Engaging students in the learning process with game-based learning: The fundamental concepts," *International Journal of Technology in Education*, vol. 4, no. 3, pp. 542–552, 2021.

[38] WestEd, "Evaluation of legends of learning games: Impact on student achievement," 2019. Accessed: 2025-04-25.

[39] C. Franzwa, Y. Tang, A. Johnson, and T. Bielefeldt, "Balancing fun and learning in a serious game design," *International Journal of Game-Based Learning*, vol. 4, no. 4, pp. 37–57, 2014.

[40] G. J. Scott Warren, "Overcoming educational game development costs with lateral innovation: Chalk house, the door, and broken window," *Journal of Applied Instructional Design*, vol. 4, no. 1, pp. 4–15, 2016.

[41] A. All. *et al.*, "Measuring effectiveness in digital game-based learning: A methodological review," *International Journal of Serious Games*, vol. 2, no. 2, pp. 35–52, 2015.

[42] M. J. Gomez *et al.*, "A systematic literature review of game-based assessment studies: Trends and challenges," *ResearchGate*, 2023.

[43] P. G. J. Irwin, *Giant Planets of Our Solar System.* Cambridge, UK: Cambridge University Press, 2009.

[44] M. M. Woolfson, "The origin and evolution of the solar system," *Astronomy & Geophysics*, vol. 41, no. 1, p. 12, 2000. Available for free access.

[45] K.-P. Schrder and R. Connon Smith, "Distant future of the sun and earth revisited," *Monthly Notices of the Royal Astronomical Society*, vol. 386, pp. 155–163, May 2008. Available for free access.

[46] nineplanets.org, "Solar system facts," Feb. 2007. origin content saved on 2015.12.12.

[47] D. Williams, "Planetary fact sheet - metric." Goddard Space Flight Center, NASA, Dec. 2021. Accessed: 2022-12-11, origin content saved on 2011.08318.

[48] NASA Space Place, "How many moons does each planet have?," Apr. 2024. origin content saved on 2024.04.21.

[49] K. D. Runyon, P. T. Metzger, S. A. Stern, and J. Bell, "Dwarf planets are planets, too: Planetary pedagogy after new horizons," *Pluto System After New Horizons (LPI Contributions)*, vol. 2133, p. 7016, 2019. Conference abstract, LPI Contribution No. 2133.

[50] J. Gradie and E. Tedesco, "Compositional structure of the asteroid belt," *Science*, vol. 216, no. 4553, pp. 1405–1407, 1982.

[51] A. L. Cochran, A.-C. Levasseur-Regourd, M. Cordiner, E. Hadamcik, J. Lasue, A. Gicquel, D. G. Schleicher, S. B. Charnley, M. J. Mumma, L. Paganini, *et al.*, "The composition of comets," *Space Science Reviews*, vol. 197, pp. 9–46, 2015.

[52] A. E. Rubin and J. N. Grossman, "Meteorite and meteoroid: New comprehensive definitions," *Meteoritics & Planetary Science*, vol. 45, no. 1, pp. 114–122, 2010.

[53] S. T. Suess, "The heliopause," *Reviews of Geophysics*, vol. 28, no. 1, pp. 97–115, 1990.

[54] P. R. Weissman, "The oort cloud," *Nature*, vol. 344, no. 6269, pp. 825–830, 1990.

[55] The Planetary Society, "The china national space administration (cnsa)," 2025. Accessed: 2025-04-27.

[56] 中国国家航天局(China National Space Administration), "重大任务(major missions)," 2025. Accessed: 2025-04-27.

[57] 中国国家航天局(China National Space Administration), "宇航产品(spaceflight products)," 2025. Accessed: 2025-04-27.

[58] N. Aeronautics and S. Administration, "Explore nasa's history," 2025. Accessed: 2025-04-27.

[59] N. Aeronautics and S. Administration, "Science missions," 2025. Accessed: 2025-04-27.

[60] E. S. Agency, "Our missions," 2025. Accessed: 2025-04-27.

[61] 屈原(Qu Yuan), 楚辞*(The Songs of Chu)*. Beijing: 中华书局(Zhonghua Book Company), 2011.

[62] E. K. Keylor, "Space vector: Video games for introductory newtonian mechanics," tech. rep., Arizona State University, 2014.

[63] S. A. Roberts and S. M. Lucas, "Evolving spaceship designs for optimal control and the emergence of interesting behaviour," in *2012 IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 342–349, IEEE, 2012.

[64] J. DSouza, "What is ai in gaming industry (40+ ai powered games in 2025)." `https://www.engati.com/blog/ai-for-gaming`, 2025. Accessed: 2025-04-23.

[65] N. Innab, Y. Liu, K. Savita, M. Shutaywi, and A. Alzahrani, "Innovative artificial intelligence and game theoretic approach for target tracking in the sensor network," *Annals of Operations Research*, 2024.

[66] K. Ljung, "Effects of field-of-view in first-person video games: A study on camera field-of-view in relation to game design," 2015.

[67] J. Byun and C. S. Loh, "Audial engagement: Effects of game sound on learner engagement in digital game-based learning environments," *Computers in Human Behavior*, vol. 46, pp. 129–138, 2015.

[68] Encyclopedia Britannica, "Newton's laws of motion." *Encyclopedia Britannica*, 2018. Original content archived on 2021-03-28.

[69] nkey, "Pull request world origin shifting in mp." `https://forums.unrealengine.com/t/pull-request-world-origin-shifting-in-mp/62649/1`, 2016. Forum post, Accessed: 2025-04-23.

[70] J. Gregory, *Game Engine Architecture*. Boca Raton: A K Peters/CRC Press, 3rd ed., 2018.