


# Game Development: Enhancing Creativity and Independent Creation in University Course

Lenka Bubenkova ✉

Department of Computers and Informatics, FEI TU of Košice, Slovakia

Emilia Pietrikova ✉ 

Department of Computers and Informatics, FEI TU of Košice, Slovakia

---

## Abstract

In this study, we tested a novel method of teaching the Unity engine to computer game design and development students. Our objective was to determine if a flexible assignment structure is the most effective for students with minimal engine experience. The study demonstrated that independent work significantly improves students' comprehension and problem-solving skills. Key findings include a 90% increase in students achieving more than the minimum required grade, a significant improvement in self-reported confidence with Unity (with 66.3% of students moving from no experience to higher skill levels), and diverse, innovative final projects that exceeded initial expectations. These results suggest that the flexible assignment approach enhances creativity and maintains high expectations for student work, ensuring their success in the game development industry. The combination of student project grades, innovative project elements, and positive feedback indicates that this method is highly beneficial and could be applied effectively in various educational settings.

**2012 ACM Subject Classification** Social and professional topics → Student assessment

**Keywords and phrases** novice programmers, assessment, learning analytics, motivation, unity engine, game development, problem-solving skills

**Digital Object Identifier** 10.4230/OASICS.ICPEC.2024.12

**Funding** This work was supported by project Kega No. 015TUKE-4/2024 “Modern Methods and Education Forms in the Cybersecurity Education”.

## 1 Introduction

In computer science and software engineering, it is essential to recognize the significance of incorporating fun and creative elements alongside traditional programming and practical tools. As computer games and other forms of digital entertainment continue to grow in popularity, educators must integrate them into their teaching methods. To remain up-to-date with the latest trends, universities must equip their students with the skills required for game development. One of the aspects of supporting this tutorial is, for example, this case study[24] of the concept of gamification used on a games development course. One practical approach to combining education and creativity is teaching students how to craft their games in the most imaginative way possible while leveraging gamification to deliver lessons through gameplay. This can be used in various courses, as described in this publication [21], where authors implemented the learning of object-oriented programming by playing computer games. While teaching students the basics of game development and the usage of engines, there is also a need to consider a theory that will help them create visually attractive and exciting games. One of the ways to develop such a game is using patterns, as is written in this publication [32], to keep players entertained and simulate and enhance reality.

Currently, there are many game engines available for creating games. Unity is a popular choice because many job opportunities require knowledge of this engine, as written in this article [17], and it is easy to use, even for students new to game development. Besides,



© Lenka Bubenkova and Emilia Pietrikova;  
licensed under Creative Commons License CC-BY 4.0

5th International Computer Programming Education Conference (ICPEC 2024).

Editors: André L. Santos and Maria Pinto-Albuquerque; Article No. 12; pp. 12:1–12:13

OpenAccess Series in Informatics



OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 12:2 Enhancing Creativity in University Course

Unity's ability to create multiplatform [13] games and projects is important for today's game development world. In addition to these aspects, the use of Unity Engine for teaching is also suitable from the point of view of the accessibility of various tutorials. As mentioned in [31], from an educational perspective, it is appropriate for students to independently find additional information about creation from professionals, programmers, and developers dedicated to creating just such instructions. In this study [8], using Unity Engine and Design-Based Research approach resulted in the high engagement of students, making an introduction to this engine smoother and more attractive. To add, based on this study [9], learning how to code by creating games promises high motivation. The aim is to teach students how to develop Unity Engine and provide enough information to continue their work even after this course.

Besides, not only game development uses Unity Engine, as we can see [19] or [34], but Unity can be used for developing various types of projects. In this example, we can see the usage of the Unity engine in the creation of the short movie "The Heretic" [10]. This provides a strong justification for teaching Unity Engine.

However, the challenge lies in making the teaching of game development exciting and engaging while encouraging students to be creative and imaginative. The traditional approach to game development can be time-consuming and challenging because students need to develop unique ideas to incorporate into their games. If students are given strict rules and guidelines to follow, they might not develop a good understanding of the engine and how to create games. As mentioned in [30], in addition to basic programming, advanced knowledge in computer graphics, databases, artificial intelligence, or, for example, physics is also necessary. Basic knowledge enables students to understand contexts better and work more effectively on assignments within the subject. Today, it is also essential to know various techniques and developments. Unity, for example, provides an excellent environment for developing virtual reality and XR in general. According to [29], the focus on the interface and, thus, the use of these XRs is extremely important for game development and projects. Each of these approaches is usable and thus feasible in the Unity Engine. Using the Unity engine for enhancing problem-solving skills and creativity for this course is also supported by outcomes of this research [26]. This study provides insights that could be applied by educators across various disciplines who wish to incorporate similar strategies to enhance engagement and learning outcomes.

### **How can creative labs improve the GameDev course?**

Up to this point, game development education in our courses has been implemented through basic versions of preexisting games, where students complete brief tasks and gain an understanding of the environment. Teaching methods are straightforward, and there is no space for considering one's elements and improvements. The entire procedure consists of downloading the game, launching it in the Unity editor, and playing it. After the game calculates their scores, students submit their scores and receive their marks. A similar approach can be seen in this [27] game made with the Unity engine.

The new method involves the entire project creation and setup process to develop a functional game similar to a flying simulator. This approach consists of three tutorials and a step-by-step guide, from creating a project to adding a user interface. The main difference between the old and new approaches is that the old approach did not allow students to create something independently. Instead, it strictly guided them through various tasks.

On the other hand, the new approach guides students through creating games, but it always leaves their final work independent of the assignment form. This means that while they must follow various steps and learn various parts, the final form is always in their

imagination. This ensures they create something unique, go through the tutorials, find more exciting and engaging parts, and work on them more. Based on the experience outlined in the article [5], the decision has been made to adopt GitLab as the submission platform for student projects.

The remainder of this paper is organized as follows: Section 2 provides background and related work. Section 3 describes our proposed approach and details the three iterations of the tutorial. Section 4 outlines the experiment setup and conduction, including our conjectures. Section 5 presents the results and discusses the findings. Finally, Section 6 concludes the paper with interpretations of the results, challenges, limitations, and suggestions for future work.

## 2 Background

With a similar course outline, this paper [14] describes the slow approach and exciting engagement for students in creating games through the Unity engine. It encourages its students to create 2D games, preparing them for future careers. We also considered the importance of active learning, as seen in this example [25], where authors used active learning based on scenarios using Unity 3D. Another engaging hands-on project experience, as described in this article [7], is using step-by-step tutorials. This kind of tutorial provides a compact way of learning and understanding problems to their core. Thinking about using Blender, we also searched for existing implementations of this tool in teaching. One of the most used approaches is the creation of educational games using Unity. This approach can introduce students to development with the Unity engine in the form of a game. With this form of teaching, as is written in this research [16] that discusses educational games in Unity, we can effectively avoid limitations to traditional teaching methods and support students' creativity.

Another approach for this tutorial is using a teacher-student model, which was optional in the previous teaching method. With the availability of a teacher during the whole process, we may support students' innovative thinking and reasoning abilities. This idea of composing into this course was supported with research [11], that analyses this Teacher-Student model.

While using the Unity engine, we were also aware of the risk of issues in students' projects, like bad smells. As was discussed in this article [6], bad smells are detectable and can be divided into categories. The main target of this course was to prevent students from developing this kind of issue. Similarly, this study [23] also discusses bad smells, and its aim was directly onto this issue in game development. This article also discusses the negative impact of bad smells and the risk that they are not always critical.

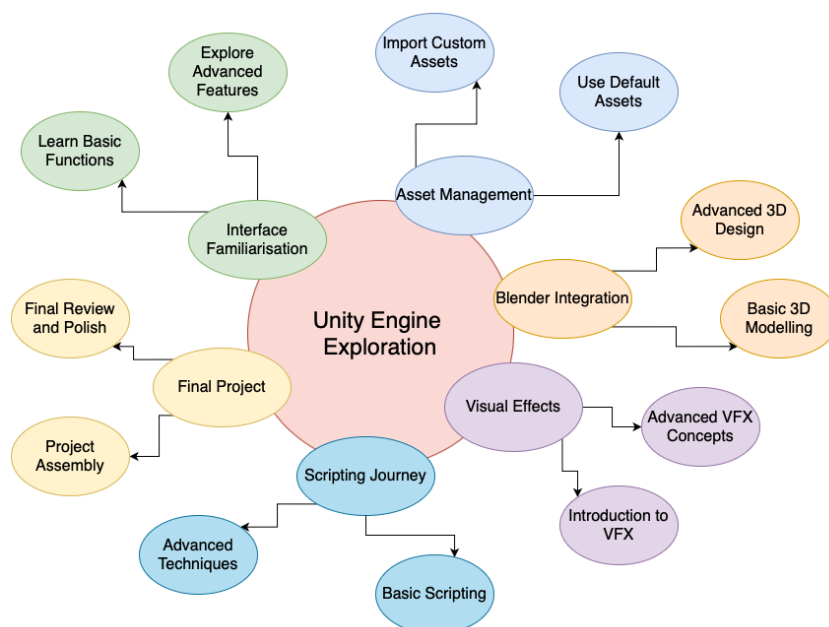
In conclusion, there is no need to push students back with their projects only because of the avoidance of bad smells. Instead, the fact that the Unity engine is fully programmable in C#, as is written in this article [33], there is a need to focus on correct and nonissue coding. However, at all times, we should be careful to avoid strictly controlling students with sets of rules to ensure creative and innovative thinking in their projects.

With the rise of immersive learning, we can also discuss the need for knowledge of the Unity engine. For example, this work [3] demonstrates the usage of Unity for the implementation of twin-screw, which is a process in the polymer. This work used Unity to implement the simulation in an interactive educational environment. A similar approach is used in the paper [35], where the problem of the informed purchase of toys is solved with the implementation of augmented reality technology using Unity 3D. Immersive interaction can also be done in full-body forms, as is described in this paper [15], where authors contributed to enabling this

kind of interaction in the metaverse. To achieve this, the authors worked with Unity 3D and other tools, successfully integrating the digital twins and immersive user experience. This work may show that students need more than the Unity engine to prepare for the industry's challenges. The usage of this information and its successful implementation in the education process is written in this article [20], where authors proved that this approach can be effective in the student's improvement in the learning process.

### 3 Proposed approach

Tutorials are created in the areas of computer game design and development. There are three tutorials. The goal of the first tutorial is to explain the engine and cover the fundamentals of project and game background creation. The second tutorial covers creating game effects using VFX Graph and explains the differences between different approaches in effect creations. The second tutorial also covers the fundamentals of the Blender[12] engine and how to create 3D models. The third tutorial served as an addition to help students add engaging components to their projects.



■ **Figure 1** The Unity Engine Learning Journey outlining the structured path from basic concepts to the final project.

Before discussion of the detailed structure of the tutorials, Figure 1 presents a comprehensive mind map of the Unity Engine Exploration course. This diagram encapsulates the learning journey from initial familiarization with the Unity interface to the final project assembly, containing advanced topics such as VFX and Blender integration.

#### 3.1 First iteration

In the beginning, students are led through the installation process. Then, they are shown how to create and run the project. This part is crucial because creating and starting with work in Unity from the total basics is fundamental for students to develop a relationship with

the engine. It makes it easier for them to understand every part that will come next from the basics. After the introduction of the engine, the first tutorial covers the creation of the game world using terrain tools. Additionally, students are encouraged to use their assets, not just those shown in the tutorial. The essential part is introducing the Asset Store and working with assets in the editor. In the future, they must understand how to create and work with assets. The next important step is adding a player. They are encouraged to pick their avatar to play their game. The exciting part of this tutorial is adding a Timeline. A Timeline is generally used to animate movements and create exciting game moments. Students are shown the basics of work with a timeline and are not given the detailed needs of using a timeline for their game; instead, they use it in their way. Using a timeline is beneficial not only for game development but also for many other fields. Short movies and ads can be created using a timeline, so this incorporation into the curriculum may widen their abilities in the future in various fields. Students are also introduced to scripting in Unity, using C#. We are teaching 3rd graders, so they must have various programming skills. Because of that, this part is mainly informative and shows them good practices in scripting in Unity. At the end of this tutorial, students have developed their game world and the basic movement of players in this world. The crucial part is personification and uniqueness. This tutorial has a role in developing interest in creations and incorporating their own elements, which may benefit their careers and development.

### 3.1.1 First iteration outcome

In the first iteration, students learned the basics of Unity and created a simple game world. For instance, one student created a terrain with mountains and a river, utilizing assets from the Unity Asset Store. In contrast, another student designed a desert landscape with custom textures.

## 3.2 Second iteration

The second tutorial was expecting better skills with unity and overall orientation of students in the engine. Firstly, there is a comparison of the two most common tools for creating visual effects in unity: VFX Graph and Particle System. Next, students moved to work with VFXGraph, creating an elemental explosion made of more layers. In this article, we can see the point of teaching students the basics of VFX [18], also with a similar approach in this article [22], as this theme is not only valid with game development but also has broad usage in many other fields. Students are asked to develop their solution in the VFX Graph editor and, later in the tutorial, find their usage in their game. This tutorial also covers the basics of working with Blender. Blender is used to create a simple target for students' games. The main reason for incorporating Blender into this course is the need for 3D models in today's game. Teaching students the basics of working with Blender and coworking with Unity may benefit project uniqueness and originality. As written in the article that discusses modeling [4], Blender was the best choice. Later in the tutorial, students get a task to create something useful with Blender and use it in their games.

### 3.2.1 Second iteration outcome

In the second iteration, students learned the differences between the Particle system and VFX Graph in the Unity engine and worked on their unique solutions. They were led to create the effect of an explosion. Students created effects that contributed to their games.

## 12:6 Enhancing Creativity in University Course

Another aspect of this iteration was the usage of Blender. This engine helped students to be more creative with assets and learn the essentials of the work in Blender. Students created various 3D models, such as players, donuts, trees, and various equipment for their games. Students also worked with VFX Graph, creating explosions and other effects, like shooting and smoke.

### 3.3 Third iteration

The third tutorial is optional. Creating effects in Unity or modeling 3D objects in Blender is only interesting for some students. Primarily, these parts are required, but the extra work on their project can be done on other developing parts. The third tutorial covers the basics of the particle system, which can be used in various ways, not only in game development, for example, like [2] or [28], the creation of UI using Canvas, and its elements, like sliders or buttons. Also, students can see the creation of levels using Scene Manager. A similar approach can be seen in this article [1], where leveling is also based on scenes in a functional 3D Unity game. A combination of UI and scene manager can create an entry screen for players to their game, and in the final project, the game will look more professional. The last part of the third tutorial is bug fixing. Students are encouraged to fix issues in existing projects to enhance their problem-solving skills and general knowledge of game development, where fixing is common and vital.

#### 3.3.1 Third iteration outcome

In the third iteration, students explored advanced features like the particle system and UI creation. For example, one student created a particle effect to simulate a magical spell, adding dynamic visual flair to their game. Another student designed a custom health bar and interactive menu, significantly improving the user experience. Additionally, one student developed a multi-level game with a main menu, level selection screen, and in-game HUD, demonstrating their ability to integrate UI elements with scene management.

### 3.4 Outcomes and Project Example

The existing project that students get to fix is available on the course page. In this project, all three tutorials are implemented, adding some unique elements and implementations to make students more curious and inspired. Students can create their own game in this project but are encouraged to create their own to solve issues that may be fixed in existing implementation and prepare them better for their future in this challenging field.

These tutorials should enhance students' independent work and support them in developing their games. Even if there are some mandatory steps for final grading, students still have the chance to add their elements and additions to their games and get the desired grade on the project. The reason is simple: game development is not a detailed, structured plan but more of an idea-adding process.

Following the detailed description of the tutorial structure, the following section clarifies the methodology utilized to evaluate its effectiveness. This analysis primarily focuses on assessing student feedback and project outcomes, which serve as critical metrics for measuring the educational impact of the implemented interventions.

## 4 Experiment setup and conduction

This study focuses on innovative and adaptable methods for teaching the fundamentals of using the Unity engine. Students' self-grading and individual development as project creators are crucial to a conclusion. This study employs a combination of self-reported data collection, enrolled students' grades, and final projects. Data from students' answers to questions were valid, and students were informed of the usage of their responses.

- Questionnaire – After grading, participants self-reported data to provide their perspectives and comments on the methodology.
- Evaluation – Using the results of final projects and their assignment grades; and also monitoring the quantity of work that exceeded the assignment's minimum.

### 4.1 Course Characteristics

This study was conducted during the 2023-2024 academic year. One hundred four third-year undergraduate students enrolled in a game development class during these academic year across both winter and summer semesters. The class included Slovak students as well as international students, and it was conducted in English and Slovak. Students had varying levels of prior experience with the Unity engine.

### 4.2 Conjectures

This paper posits several conjectures to evaluate the efficacy of a novel instructional approach in Unity engine development. These conjectures are implanted in the conviction that structured yet flexible learning environments can significantly enhance students' educational experience and outcomes, irrespective of their prior technical experience.

The research process involved three main iterations, each designed to build students' skills and creativity progressively. The first iteration focused on fundamental Unity skills, the second on advanced features like VFX and Blender, and the third on optional enhancements such as UI and bug fixing.

The following hypotheses are formulated to rigorously test the effectiveness of these educational methodologies through empirical evidence and data-driven analysis:

- ▶ **Conjecture 1.** *Students do not have to have prior skills and an introduction to Unity engine development to complete this course.*
- ▶ **Conjecture 2.** *If students are given more freedom in their assignments, they will be more creative and driven to work on their projects, which will ultimately result in better grades.*
- ▶ **Conjecture 3.** *After working more independently, students feel secure in their knowledge.*
- ▶ **Conjecture 4.** *Students will include unique elements and interesting strategies of their own.*

To verify these conjectures, employment of the various methods was used: Conjectures 1 and 2 were assessed using a structured questionnaire that studied student engagement and learning outcomes. For Conjecture 3, an analysis of existing student performance data was conducted to determine the impact of independence on learning security. Conjecture 4 was evaluated through project reviews that assessed the creativity and uniqueness of student submissions.

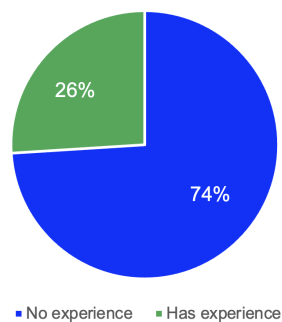
## 5 Results

The examination of students' results, final projects, and self-reported information from questionnaires was essential to analyze the effectiveness of the teaching method. This section presents the findings of these analyses, which help validate the conjectures proposed in this paper. An overview of the analyzed data and a summary of the findings related to the hypotheses are provided below.

### 5.1 Conjecture 1

In the following Figure Figure 5.1, it is visible that the number of students with some experience before attending this course was much smaller than those that had not previously worked with the Unity engine. Percentually, only 26% worked with the engine. This section aims to determine if these students will experience any resulting disadvantages.

Student's experience with Unity Engine



■ **Figure 2** Student's experience with Unity Engine before tutorials.

The Pearson correlation coefficient was used to examine whether prior experience with the Unity Engine statistically impacts the grading scores. The test yielded a coefficient of 0.12, considered a low degree of correlation. This result suggests no statistically significant score difference between individuals with and without previous experience using the Unity Engine. Consequently, previous knowledge of Unity only significantly influences the observed grading outcomes.

This proved Conjecture 1, where was supposed that students do not have to have prior knowledge to perform outstanding results in this course.

### 5.2 Conjecture 2

Immediately after the personal defense, students were awarded points for the developed projects based on a predetermined evaluation on the subject page at the end of each tutorial. The attached table Table 1 shows that the highest rating was 10 points, representing the maximum of the possible points obtained. Completing all the necessary steps in the tutorial could obtain the minimum number of points, which was 6. The result, which was 10 points, means that students did extra work on this project and added many particular tasks from all the tutorials. This point evaluation was obtained by more than half of all students. The second highest rating belongs to students with the same score, 14 for both 8 and 9 points. At the same time, a low assessment of the assignment occurred in only one case, in the form of one point. Therefore, completing the instructions by the students is exceptionally successful.



In the questionnaire, students had to answer the question, “Do you think that your point evaluation corresponds to the time you devoted to the assignment?” where the answer “Yes” reached the value of 94.2%, and the answer “No, I devoted a lot of time to this assignment” had a value of 3.5%, which can be considered as the result of a fair assessment and student satisfaction with this aspect of the university courses as well.

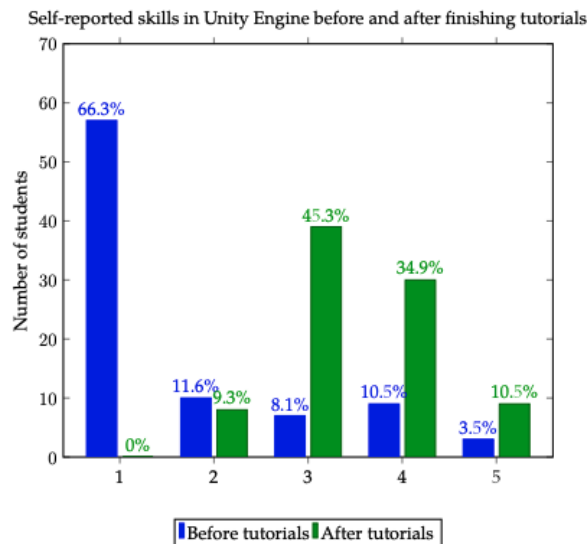
■ **Table 1** Table for students grades.

points	0p	1p	2p	3p	4p	5p	6p	7p	8p	9p	10p
students	0	1	0	0	3	3	3	7	14	14	55

This proved Conjecture 2, where was supposed that students would be motivated to work harder on their assignments if they were given some space for their creations. This also proved that 90% of students did more than the required work, graded with 6 points.

### 5.3 Conjecture 3

Students also had the opportunity to answer questions about their skills with Unity Engine before and after completing tutorials. As can be seen on the attached chart Figure 5.3, self-rating skills in Unity Engine, from rating to scale from 1 to 5, where 1 represents zero experience and five the highest level of ability properties, all students managed to move to a higher rating. In particular, the drop in the change from 66.3% to 0% in experience indicates excessive success. On the second side, self-assessment in the highest degree rose by 7%, assessment in the form of 4 out of 5 increased by 24.4%, and the rating on the 3 out of 5 scale increased by 37.2%.



■ **Figure 3** Self-reported skills in Unity Engine before and after finishing tutorials.

This proved Conjecture 3, where was supposed that students would feel more secure in their knowledge if they worked independently.

## 5.4 Conjecture 4

The students were instructed to create and add their elements independently. After collecting and evaluating the projects, it was evident that the students demonstrated fascinating diversity and creativity in their work. Each student's individuality was visible in the creation of their assignments. One exciting element was the use of Timeline in the form of game animation, which was used effectively in several projects. For example, a cinematic animation of a military plane flying over the sea and observing various elements such as ships, houses, and the sea. The manual mainly covers movement in the form of flying, but the resulting student projects brought many new solutions, such as walking. This means that encouraging the students to create and experiment on their own was enough, and when they were left to their own will, various game projects were created.

This proved Conjecture 4, where was expected that students would create unique and personal projects with various elements over the course outline.

## 6 Conclusion

The tutorial outcomes demonstrate considerable advancements in students' capabilities to produce creative outputs and solve complex problems. This section will analyze the implications of these results, contrasting them with traditional educational methodologies in game design and examining their potential transformative impacts on the curriculum.

### 6.1 Interpretation of Results

A notable aspect of the student projects was their diversity in game design approaches. While the core tutorials were designed around constructing simulations for flight simulators and first-person shooters, many students ventured beyond these confines. They crafted unique game environments, where characters navigated through complex worlds or diverged completely to create cinematic short animations. These projects not only stuck to the foundational elements of the tutorials but also incorporated innovative modifications and additions. For instance, Figure 6.1 illustrates a project featuring an interactive walking character, demonstrating the practical application of the skills acquired through the course. Positive feedback regarding the final projects was on the availability of project flying-heroes, where were implemented all three tutorials with added addins and unique elements. As was said in the student feedback, the ability to see the existing games and their parts was beneficial, not only for students to see how the parts of the tutorial are implemented but also as an inspiration for their projects.



■ **Figure 4** Example of a student-developed game showcasing an interactive character.

## 6.2 Challenges and Limitations

One primary challenge was guiding students through managing extensive projects and utilizing comprehensive file systems within their version control repositories. Furthermore, the affiliate's familiarity with Unity Engine was initially low, posing a significant learning curve. However, most issues were swiftly addressed through online resources or direct interventions during lectures, enhancing the learning experience and problem-solving efficiency.

## 6.3 Future work

Future work on this course should contain more detailed explanations for problematic parts written in student responses. Also, students asked for additional content, such as AI in the form of enemies, more scripting in the course, or video tutorials, as they thought they would benefit more from them while working at home without the lecturer's presence. Detailed explanations of problematic topics may help them better adapt to future needs. Besides this, this course is up to date and should be sustainable for more years in the future, but after some time, some additions and work would be needed as the game development world moves further.

The study explores the integration of game design and development courses at the university, highlighting the inclusion of subjects like Unity Engine and Blender, which are popular among students and offer practical skills demanded in the industry. These courses introduce students to complex tools like effects, animations, interfaces, and advanced features like Timeline and VFX Graph.

The curriculum is designed not merely to engage students during lectures but to inspire independent work and creativity in their projects. This approach has proven effective, with students delivering highly successful projects that incorporate their unique solutions. The addition of Blender enriches the creative potential and deepens students' interest in game development and game design.

Student feedback regarding satisfaction with the course and the instructions' clarity has been overwhelmingly positive. The results met and exceeded initial expectations, with students developing innovative solutions promptly. Future enhancements include AI enemies, expanded scripting options, and multimedia instructional materials, reflecting the students' desire for more comprehensive and prolonged engagement with game development tools.

It can be considered, that using this approach in various courses can help to enhance students' learning and adaptability across different disciplines. This method may be integrated into other areas to foster technical skills, critical thinking, problem-solving, and creative design thinking. By implementing these techniques widely, educational institutions can better prepare students for the complexities of modern professional environments while equipping them with the necessary tools to excel in their chosen fields.

---

## References

- 1 Shanmuk Srinivas Amiripalli, Mukkamala SNV Jitendra, Surendra Talari, Sannith Akkireddi, and D Sateesh Kumar. Design and implement an artificial intelligence based zombie's application using unity3d, 2020.
- 2 Natalia Ampilova, Igor Soloviev, and Michail Syasko. Computer modeling the effect of weak electromagnetic field on charged particles by unity engine. In *2020 International Conference and Exposition on Electrical And Power Engineering (EPE)*, pages 082–086, 2020. doi:10.1109/EPE50722.2020.9305579.

- 3 Pedro Santos Bartolomé, Daniel Just, Ariana Bampouli, Simon Kemmerling, Aleksandra Buczko, and Tom Van Gerven. Immersive learning through simulation: implementing twin screw extrusion in unity. In Antonios C. Kokossis, Michael C. Georgiadis, and Efstratios Pistikopoulos, editors, *33rd European Symposium on Computer Aided Process Engineering*, volume 52 of *Computer Aided Chemical Engineering*, pages 3489–3494. Elsevier, 2023.
- 4 Iliyyar Bikmullina and Enzhe Garaeva. The development of 3d object modeling techniques for use in the unity environmen. In *2020 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon)*, pages 1–6, 2020. doi:10.1109/FarEastCon50210.2020.9271568.
- 5 Miroslav Binas. Version control system in cs1 course: Practical experience. In *2013 IEEE 11th International Conference on Emerging eLearning Technologies and Applications (ICETA)*, October 2013. doi:10.1109/ICETA.2013.6674398.
- 6 Antonio Borrelli, Vittoria Nardone, Giuseppe A. Di Lucca, Gerardo Canfora, and Massimiliano Di Penta. Detecting video game-specific bad smells in unity projects. In *Proceedings of the 17th International Conference on Mining Software Repositories, MSR '20*, pages 198–208, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3379597.3387454.
- 7 Simon Bouvier-Zappa, Olivier Dionne, and David Hunt. Advanced use cases for animation rigging in unity. In *ACM SIGGRAPH 2019 Studio*, SIGGRAPH '19, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3306306.3328748.
- 8 Oswald Comber, Renate Motschnig, Hubert Mayer, and David Haselberger. Engaging students in computer science education through game development with unity. In *2019 IEEE Global Engineering Education Conference (EDUCON)*, pages 199–205, 2019. doi:10.1109/EDUCON.2019.8725135.
- 9 Oswald Comber, Renate Motschnig, Hubert Mayer, and David Haselberger. Engaging students in computer science education through game development with unity. In *2019 IEEE Global Engineering Education Conference (educon)*, pages 199–205. IEEE, 2019.
- 10 Veselin Efremov and Adrian Lazar. Real-time procedural vfx characters in unity's real-time short film "the heretic". In *ACM SIGGRAPH 2019 Real-Time Live!*, SIGGRAPH '19, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3306305.3332363.
- 11 Hui Fang, Hongmei Shi, Jiuzhou Zhang, and Marimuthu Karuppiah. Effective college english teaching based on teacher-student interactive model. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 22(3), March 2023. doi:10.1145/3486676.
- 12 Lance Flavell. *Beginning Blender: Open Source 3D Modeling, Animation, and Game Design*. Apress, USA, 1st edition, 2010.
- 13 Maxwell Foxman. United we stand: Platforms, tools and innovation with the unity game engine. *Social Media+ Society*, 5(4):2056305119880177, 2019.
- 14 Christopher L. Hideg and Debatosh Debnath. A programming course using video game design with platform projects. In *2018 IEEE International Conference on Electro/Information Technology (EIT)*, pages 0030–0034, 2018. doi:10.1109/EIT.2018.8500103.
- 15 Shimasadat Hosseini, Ali Abbasi, Luis G. Magalhaes, Jaime C. Fonseca, Nuno M.C. da Costa, António H.J. Moreira, and João Borges. Immersive interaction in digital factory: Metaverse in manufacturing. *Procedia Computer Science*, 232:2310–2320, 2024. 5th International Conference on Industry 4.0 and Smart Manufacturing (ISM 2023). doi:10.1016/j.procs.2024.02.050.
- 16 Zhiyong Hu, Qing Xu, and Guang Huang. Discussion on educational games based on unity. In *Proceedings of the 2022 6th International Conference on Education and E-Learning, ICEEL '22*, pages 67–74, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3578837.3578847.
- 17 Afzal Hussain, Haad Shakeel, Faizan Hussain, Nasir Uddin, and Turab Latif Ghouri. Unity game development engine: A technical survey. *Univ. Sindh J. Inf. Commun. Technol*, 4(2):73–81, 2020.

- 18 Manolya Kavakli and Cinzia Cremona. The virtual production studio concept – an emerging game changer in filmmaking. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 29–37, 2022. doi:10.1109/VR51125.2022.00020.
- 19 Liuxian Li and Zhiyang Fang. Earthquake escape simulator: A system for disaster knowledge popularization. In *Journal of Physics: Conference Series*, volume 2333, page 012002. IOP Publishing, 2022.
- 20 Xiaoxiao Liu, Yiming Shen, Yukari Nagai, and Hirokazu Kato. Use of a mixed-reality creative environment in design education. *Computers & Education: X Reality*, 4:100055, 2024.
- 21 Jakub Livovský and Jaroslav Porubán. Learning object-oriented paradigm by playing computer games: Concepts first approach. *Open Computer Science*, 2014. doi:10.2478/s13537-014-0209-2.
- 22 Jonathan Mortimer. How universities can better engage with the animation/vfx sector in scotland. *Animation Practice, Process & Production*, 7(1):157–173, 2018.
- 23 Vittoria Nardone, Biruk Muse, Mouna Abidi, Foutse Khomh, and Massimiliano Di Penta. Video game bad smells: What they are and how developers perceive them. *ACM Trans. Softw. Eng. Methodol.*, 32(4), May 2023. doi:10.1145/3563214.
- 24 Siobhan O’Donovan, James Gain, and Patrick Marais. A case study in the gamification of a university-level games development course. In *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference, SAICSIT ’13*, pages 242–251, New York, NY, USA, 2013. Association for Computing Machinery. doi:10.1145/2513456.2513469.
- 25 Hyesung Park, Sean Yang, and Hongsik Choi. Scenario based active learning programming with unity 3d. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education, SIGCSE ’20*, page 1283, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3328778.3372582.
- 26 Banyapon Poolsawas and Winyu Niranatlamphong. Using a game development platform to improve advanced programming skills. *Journal of Reviews on Global Economics*, 6:328–334, 2017.
- 27 Vincent Schiller. Enc#ypted: An educational game for programming in the unity engine. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems, CHI EA ’21*, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3411763.3451852.
- 28 Jasmine Y. Shih, Kalina Borkiewicz, AJ Christensen, and Donna Cox. Interactive cinematic scientific visualization in unity. In *ACM SIGGRAPH 2019 Posters, SIGGRAPH ’19*, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3306214.3338588.
- 29 Branislav Sobota. *Computer Game Development*. IntechOpen, Rijeka, August 2022. doi:10.5772/intechopen.97983.
- 30 Branislav Sobota and Emília Pietriková. *Computer Science for Game Development and Game Development for Computer Science*. IntechOpen, Rijeka, November 2023. doi:10.5772/intechopen.1000364.
- 31 Branislav Sobota and Emília Pietriková. The role of game engines in game development and teaching. In Branislav Sobota and Emília Pietriková, editors, *Computer Science for Game Development and Game Development for Computer Science*, chapter 5. IntechOpen, Rijeka, 2023. doi:10.5772/intechopen.1002257.
- 32 Branislava Vranić and Valentino Vranić. Patterns of recreating reality in games. In *Proceedings of 29th Conference on Pattern Languages of Programs, PLoP*, 2022.
- 33 Ursula Wolz, Gail Carmichael, and Chris Dunne. Learning to code in the unity 3d development platform. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education, SIGCSE ’20*, page 1387, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3328778.3367010.
- 34 Haolong Yang, Chunqiang Hu, Guwei Li, and Jingchun Fan. A fire escape simulation system based on the dijkstra algorithm. *Comput. Syst. Sci. Eng.*, 39(3):365–372, 2021.
- 35 Lingxin Yu, Jiacheng Zhang, Xinyue Wang, Siru Chen, Xuehao Qin, Zhifei Ding, and Jiahao Han. Constructing immersive toy trial experience in mobile augmented reality. *Internet of Things and Cyber-Physical Systems*, 4:250–257, 2024. doi:10.1016/j.iotcps.2024.02.001.