JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 35 (2025) 197–201
Received: 15 February 2025
Accepted: 23 April 2025

Comparative analysis of the performance of Unity and Unreal Engine

Analiza porównawcza wydajności silnika Unity i Unreal Engine

Robert Kilijanek*, Marek Miłosz

Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract

The article analyzes the performance of the two most popular game engines: Unity and Unreal Engine. The performance analysis focuses on examining the load on selected components during physics simulations, as well as the rendering of static and dynamic objects. The aim of the research is to identify the more efficient engine based on the selected tests. The experiments included physics simulations of spheres, tree rendering, and character rendering in an urban environment. The results indicate which engine achieved lower and more stable hardware resource usage. The conclusions provide developers with guidance on choosing the appropriate engine based on the project's specifications.

Keywords: Unity; Unreal Engine; performance; rendering

Streszczenie

Artykuł analizuje wydajność dwóch najpopularniejszych silników do gier: Unity i Unreal Engine. Analiza wydajności skupia się na badaniu obciążeń wybranych komponentów podczas symulacji fizyki oraz renderowania obiektów statycznych i dynamicznych. Celem badań jest określenie wydajniejszego silnika na podstawie przeprowadzonych testów. Eksperymenty obejmowały symulacje fizyki kulek, renderowanie drzew oraz postaci w środowisku miejskim. Wyniki wskazują, który z silników charakteryzuje się mniejszym i bardziej stabilnym zużyciem zasobów sprzętowych. Na podstawie wniosków producenci mogą dobrać odpowiedni silnik, uwzględniając specyfikę projektu.

Słowa kluczowe: Unity; Unreal Engine; wydajność; renderowanie

*Corresponding author

Email address: **kilijanek.robert@gmail.com** (R. Kilijanek)

1. Introduction

The gaming industry generates increasing profits each year. In 2024, the industry's revenue reached 184 billion USD [1]. Among the released titles are games from independent developers, as well as productions from the AAA (high-budget) segment. Every developer uses tools to create games that efficiently manage system resources. The market offers numerous engines designed for efficient game development. Two of the most popular game engines are Unity and Unreal Engine. Unity was developed by Unity Technologies [2]. Unreal Engine, on the other hand, is a game engine created and maintained by Epic Games [3]. These engines have been available for many years and are continuously evolving, introducing increasingly advanced tools. Additionally, both engines set trends in the gaming industry by implementing new solutions and technologies. The purpose of this study is to determine which engine, Unity or Unreal Engine offers better performance based on prepared scenarios.

1.1. Literature Review

Unity and Unreal Engine are two of the most recognizable and widely used engines in the gaming industry [4]. Their efficiency and development have been the subject of numerous studies. Unity was developed by Unity Technologies and gained popularity due to its intuitive interface and access to a vast library of ready-made assets (pre-made 3D models, textures, sounds, and scripts), created by both developers and the company itself [2]. These assets, available in both free and paid versions, include ready-to-use 3D models, textures, and scripts, which significantly accelerate project development. Thanks to its numerous advantages, Unity is chosen by both large developers and independent creators. Unity is most commonly used for mobile applications due to its ability to run efficiently with limited resources [5].

Unreal Engine, developed by Epic Games, is an advanced engine offering high-quality graphics [3]. Its standout feature is its suite of tools that enable advanced graphical rendering. With these tools, developers can create realistic scenes and detailed environments [6]. Due to its numerous advanced features, Unreal Engine is an ideal solution not only for AAA games but also for providing versatile tools for independent creators.

Both Unity and Unreal Engine continuously evolve their platforms to meet the demands of developers and the expectations of players. In recent years, the development of VR/AR technologies has become particularly significant, as these technologies are becoming increasingly advanced and represent a growing trend in the market [7].

Unity and Unreal Engine offer high performance and flexibility, but differences in the quality of applications arise depending on the target platform due to the tools they provide. It is noticeable that Unity performs better with cross-platform applications, while Unreal Engine achieves significantly better results than Unity on desktop platforms [8].

Vohera and others analyzed game engine architectures, comparing four popular environments: Unity, Unreal Engine, CryEngine, and GameMaker. The research examined key functionalities, including the rendering engine, physics system, animation module, sound system, and scripting system. The authors highlighted fundamental differences in the technologies used by each engine. Additionally, the study outlines the supported programming languages and licensing models, emphasizing their impact on development. The authors conclude that the optimal engine choice depends on the specific requirements of the project [9].

Zhang conducted research on the Particle Effects system in Unreal Engine, focusing on techniques for creating and optimizing visual effects. The study details the process of generating a snow effect, utilizing Unreal Engine's modular system. This system allows developers to add, remove, and modify modules as needed, providing greater flexibility in designing complex and dynamic visual effects. By leveraging this approach, Unreal Engine enhances both performance and creative possibilities in effect generation [10].

Salama and Elsayed's study analyzes the performance of Unity and Unreal Engine under real-time conditions. The research examines FPS stability, performance under heavy load, and resource loading time. The findings indicate that Unreal Engine outperforms Unity in high-quality graphics and complex physics scenarios, while Unity is more efficient in simpler scenes due to lower hardware demands [11].

1.2. Hypothesis

The research hypothesis assumes that **Unreal Engine demonstrates higher performance than Unity in scenarios requiring complex graphical rendering and advanced physics calculations**. Additionally, it is expected that Unreal Engine, with its numerous advanced tools for graphic optimization, performs better in scenarios involving a significant number of objects. Conversely, Unity may prove to be more efficient and effective in simpler, less complex scenarios where the number of objects is limited.

2. Materials and Methods

2.1. Research Environment

The research focused on analyzing the performance of the two most popular game development engines: Unity and Unreal Engine. The experiments were conducted across three different scenarios, including physics simulation and rendering of static and dynamic objects. During each scenario, measurements were taken of CPU, GPU, and RAM usage, as well as FPS (Frames per Second). The scenarios were designed to replicate real-world situations that developers may encounter during the game development process. For each scenario, textures were prepared to be as similar as possible, and identical starting parameters were applied, ensuring comparable results.

The tests were conducted on a machine running Windows 10 Pro version 22H2. The Unity tests were performed using Unity 2021.3.26f1 LTS, while the Unreal Engine tests were conducted with Unreal Engine 4.27.2.

2.2. Object of the Study

The tests were conducted on a computer equipped with an Intel Core i5-4460 processor, a quad-core CPU with a clock speed of 3.2 GHz, and an MSI GeForce GTX 1050Ti graphics card with 4 GB of GDDR5 memory. Additionally, the system had 8 GB of DDR3 RAM and a 256 GB SSD, ensuring fast data access during testing. This hardware configuration represented a mid-range computer, closely resembling the type of equipment used by both developers and end users, such as gamers.

To monitor and record hardware resource usage, such as CPU, GPU, and RAM consumption, MSI Afterburner and HWmonitor were employed. Additionally, FPS was monitored and recorded using Fraps. These tools allowed for real-time tracking and logging of resource usage, facilitating the early detection of any anomalies.

2.3. Research Procedure

As part of the research procedure, three different test scenarios were prepared and conducted. Each scenario reflects the most commonly encountered aspects during the game development process.

The first scenario involved a physics simulation – the sphere variant. A transparent rectangular prism was prepared, inside which small pre-made spheres bounced around. This scenario allowed for the evaluation of how each engine handled physics simulations and managed multiple collisions, both between the spheres and the rectangular prism as well as between the spheres themselves. Three test sets were prepared, featuring 100, 500, and 1000 spheres, respectively. Each sphere was given an initial impulse and a starting point. During the scenario, CPU usage, GPU usage, RAM usage, and FPS were recorded and monitored.

The second scenario focused on rendering static objects. A large space was prepared, on which 3D tree models were randomly generated from ready-made assets for Unity [12]. Similarly, 3D tree models were randomly generated from ready-made assets for Unreal Engine [13].Trees were generated in a non-overlapping manner, and global lighting was prepared. Three test sets were prepared, featuring 100, 500, and 1000 trees, respectively. Each scenario was monitored to record CPU usage, GPU usage, RAM usage, and FPS. The setup allowed for evaluating how each engine handled the rendering of large numbers of static objects and global lighting.

The last scenario involved rendering dynamic objects in a static environment. A city segment was prepared using ready-made assets, consisting of several buildings and typical objects commonly found in urban spaces for Unity [14]. Similarly, a city segment was created using ready-made assets, featuring the same types of buildings and objects for Unreal Engine [15]. Within this city segment, 3D character models moved randomly from ready-made assets for Unity [16]. Similarly, 3D character models moved randomly from ready-made assets for Unreal Engine [17]. The characters navigated within a designated area of the city using an AI script, interacting with their surroundings and other characters through collisions. Three test sets were prepared, featuring 10, 25, and 50 NPCs (Non-Player Characters), respectively. During the scenario, CPU usage, GPU usage, RAM usage, and FPS were recorded and monitored. This setup enabled an assessment of how each engine handled dynamic object rendering and AI-driven movement in a static environment.

Each scenario was monitored and recorded multiple times to ensure reliable results. After recording, the data was cleaned to remove any outliers. The results from each test were then averaged to make the outcomes comparable. This process allowed for drawing conclusions and identifying the advantages of one engine over the other in each scenario.

Each test was repeated three times to ensure reliability and minimize the impact of random fluctuations. The results were then averaged to provide a clear comparison between the performance of both engines.

3. Research Results and Discussion

3.1. Physics Simulation – the Sphere Variant

The results of the sphere variant for 100, 500, and 1000 spheres revealed noticeable performance differences between Unity and Unreal Engine.

The sphere variant revealed (Table 1) that Unity placed less strain on the GPU, particularly noticeable with 1000 spheres, where GPU usage was 11.76%. However, this came at the cost of a significant drop in FPS, with an average of 171.46 FPS. In contrast, Unreal Engine demonstrated higher GPU usage, reaching up to 46.35% with the 100-sphere variant, but provided greater stability, maintaining an average of 187.38 FPS. Unity managed resources more efficiently under heavy loads, while Unreal Engine ensured more stable frame rates during rendering.

Table 1: Measurement Results for Unity and Unreal Engine for Sphere Variant

Type of Engine	Measured Component	Sphere Variant		
		100	500	1000
Unity	CPU	64.81%	62.35%	66.14%
	GPU	25.72%	23.29%	11.76%
	RAM (MB)	8494	8339	8455
	FPS	188.45	188.45	171.76
Unreal Engine	CPU	73.92%	70.81%	71.04%
	GPU	46.35%	41.04%	29.56%
	RAM (MB)	7977	7957	8599
	FPS	187.38	184.44	182.54

3.2. Rendering of Static Objects

The results of rendering static objects for 100, 500, and 1000 trees showed significant performance differences between Unity and Unreal Engine.

The results of static object rendering (Table 2) showed that Unity had lower GPU usage, ranging between 12.48% and 13.76%. RAM usage was also lower compared to Unreal Engine, and Unity maintained stable frame rates at 184 FPS. Unreal Engine exhibited higher GPU usage but achieved slightly higher FPS. Unity demonstrated more efficient resource management, whereas Unreal Engine delivered superior graphical performance.

Table 2: Measurement Results for Unity and Unreal Engine for Tree Variant

Type of Engine	Measured Component	Tree Variant		
		100	500	1000
Unity	CPU	45.58%	67.33%	58.78%
	GPU	13.76%	13.42%	12.48%
	RAM (MB)	7767	8605	7915
	FPS	186.54	184.58	184.58
Unreal Engine	CPU	75%	71.73%	71.19%
	GPU	41.66%	48.76%	55.19%
	RAM (MB)	8309	8212	8362
	FPS	188.96	188.96	182.98

3.3. Rendering of Dynamic Objects

The results of rendering dynamic object for 10, 25, and 50 NPCs revealed noticeable performance differences between Unity and Unreal Engine (Table 3).

Table 3: Measurement Results for Unity and Unreal Engine for NPCs

Type of Engine	Measured Component	Number of NPCs		
		10	25	50
Unity	CPU	59.95%	61.67%	63.18%
	GPU	23.12%	30.22%	32.95%
	RAM (MB)	7833	8028	7945
	FPS	181.38	179.44	184.62
Unreal Engine	CPU	58.17%	87.28%	82.93%
	GPU	31.88%	67.35%	55.65%
	RAM (MB)	8002	7798	7822
	FPS	184.59	182.01	180.55

In the scenario involving the rendering of dynamic objects (Table 3), Unity demonstrated lower GPU usage, averaging 32.95%, more stable RAM usage, and consistent frame rates for 50 NPCs. Unreal Engine made greater use of the GPU, with the highest average reaching 67.35% for 25 NPCs, but maintained more stable frame rates. While Unreal Engine managed resources better with a larger number of characters, Unity exhibited greater overall resource stability.

4. Limitations

Despite the rigorous methodology applied in this study, several limitations should be acknowledged, as they may have influenced the obtained results and should be considered when interpreting the findings.

The study was conducted on a single hardware configuration featuring an Intel Core i5-4460 processor, an MSI GeForce GTX 1050Ti graphics card, 8 GB of DDR3 RAM, and a 256 GB SSD. While this represents a mid-range system, performance may vary significantly on high-end or lower-end hardware. The results might not fully reflect performance differences on modern GPUs, CPUs with higher core counts, or systems with different memory architectures.

The tests were performed using Unity 2021.3.26f1 LTS and Unreal Engine 4.27.2. Since both engines are continuously updated, introducing optimizations and new features, the findings may become outdated with future releases. Changes in rendering techniques or resource management in newer versions could alter the observed performance characteristics.

5. Future Work

This study focused on comparing Unity and Unreal Engine in specific test scenarios, but several areas remain for further research.

Future studies could examine performance across diverse hardware configurations, including high-end GPUs, multi-core CPUs, and varying RAM capacities. Testing on mobile and console platforms would also provide insights into cross-platform optimization.

Another key area is evaluating newer engine versions, as updates may introduce performance improvements and new rendering techniques like real-time ray tracing.

Expanding to more complex scenarios, such as AI processing, open-world environments, and multiplayer networking, could reveal additional differences in engine efficiency. Investigating multi-threading and memory management would further enhance understanding of resource allocation.

Finally, future research could assess workflow efficiency and ease of development, considering factors like project setup, debugging tools, and asset integration**.** Addressing these aspects would provide a more comprehensive comparison, helping developers choose the right engine for their needs.

6. Conclusions

Unreal Engine proved to be more effective in complex scenarios, such as rendering static and dynamic objects with a large number of elements on the scene. In these scenarios, it achieved higher FPS stability thanks to its advanced graphical tools. This supports the hypothesis that Unreal Engine demonstrates superior performance in complex scenarios.

Unity exhibited greater efficiency in less complex scenarios through lower GPU and RAM usage. Additionally, it managed resources more effectively, supporting the hypothesis that Unity is better suited for projects with significantly lower complexity.

The results confirm the advantage of Unreal Engine in large, graphically complex projects and the efficiency of Unity in less demanding scenarios.

References

[1] Games industry made $184.3 billion in 2024 – consoles segment was 84% digital, TweakTown,, https://www.tweaktown.com/news/102255/games-industry-made-184-3-billion-in-2024-consoles-segment-was-84-digital/index.html/, [07.01.2025].

[2] Unity – Real-Time Development Platform, Unity Technologies, https://unity.com/, [26.12.2024].

[3] Unreal Engine – Real-Time 3D Creation Tool, Epic Games, https://www.unrealengine.com/en-US, [26.12.2024].

[4] Comparing Popular Game Engines, PubNub Blog, https://www.pubnub.com/blog/comparing-popular-game-engines/, [29.01.2025].

[5] P. Mishra, U. Shrawankar, Comparison between famous game engines and eminent games, Int. J. Interact. Multimed. Artif. Intell. 4 (2016) 69–77, https://doi.org/10.9781/ijimai.2016.4113.

[6] J. Lee, Ł. Piwko, K. Gotlin, Unreal Engine: Nauka pisania gier dla kreatywnych, Helion, Gliwice, 2017.

[7] K. Abramowicz, P. Borczuk, Comparative analysis of the performance of Unity and Unreal Engine game engines in 3D games, J. Comput. Sci. Inst. 30 (2024) 53–60, https://doi.org/10.35784/jcsi.5473.

[8] E. Christopoulou, S. Xinogalos, Overview and comparative analysis of game engines for desktop and mobile devices, Int. J. Serious Games 4(4) (2017) 21–36, https://doi.org/10.17083/ijsg.v4i4.194.

[9] C. Vohera, H. Chheda, D. Chouhan, A. Desai, V. Jain, Game engine architecture and comparative study of different game engines, In 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), IEEE (2021) 1–6, https://doi.org/10.1109/ICCCNT51525.2021.9579618.

[10] J. Zhang, Implementation and optimization of particle effects based on Unreal Engine 4, J. Phys. Conf. Ser. 1575 (2020) 012187, https://doi.org/10.1088/1742-6596/1575/1/012187.

[11] R. Salama, M. Elsayed, A live comparison between Unity and Unreal game engines, Glob. J. Inf. Technol. Emerg. Technol. 11(1) (2021) 1–7, https://doi.org/10.18844/gjit.v11i1.5288.

[12] Realistic Tree 9 – Rainbow Tree, Unity Asset Store, https://assetstore.unity.com/packages/3d/vegetation/trees/realistic-tree-9-rainbow-tree-54622, [05.09.2024].

[13] Open World Demo Collection, Unreal Engine Marketplace, https://www.unrealengine.com/marketplace/en-US/product/open-world-demo-collection, [09.09.2024].

[14] Simple City Pack – Plain, Unity Asset Store, https://assetstore.unity.com/packages/3d/environments/urban/simple-city-pack-plain-100348, [15.09.2024].

[15] Modular Military Operation Urban Training Environment – Civilian Pack, Unreal Engine Marketplace, https://www.unrealengine.com/marketplace/en-US/product/modular-military-operation-urban-training-environment-civilian-pack, [20.09.2024].

[16] Low Poly Soldiers Demo, Unity Asset Store, https://assetstore.unity.com/packages/3d/characters/low-poly-soldiers-demo-73611, [10.07.2024].

[17] 3D NPC Models – Virtus Learning Hub, VirtusHub, https://virtushub.com/courses/resources/lectures/39264137, [20.07.2024].