JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 20 (2021) 247–253
Received: 27 June 2021
Accepted: 9 July 2021

Comparative analysis of Unity and Unreal Engine efficiency in creating virtual exhibitions of 3D scanned models

Analiza porównawcza wydajności silników Unity i Unreal Engine w aspekcie tworzenia wirtualnych pokazów modeli pochodzących ze skanowania 3D

Agata Malwina Ciekanowska*, Adam Krzysztof Kiszczak – Gliński*, Krzysztof Dziedzic

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

Abstract
The main purpose of this work was to compare two game engines (Unreal Engine and Unity) in creating virtual exhibitions. The article is a scientific description of a test of their efficiency. For the needs of the research two identical test applications built on the basis of the same assets were created. Those applications enabled researchers to examine and compare the efficiency of engines in question, as well as familiarizing themselves with the workflow on each platform. The comparative analysis of gathered data let more effective solution to emerge, which happens to be Unity engine.

Keywords: virtual museum; 3D models; game engines; comparative analysis

Streszczenie
Głównym celem tej pracy było porównanie dwóch silników gier (Unreal Engine oraz Unity) w tworzeniu wirtualnych pokazów. W artykule opisano doświadczenie badające ich wydajność. Na potrzeby eksperymentu przygotowano dwie bliźniacze aplikacje testowe, zbudowane na bazie tych samych assetów. Pozwoliły one na zbadanie i porównanie wydajności rozpatrywanych silników oraz zapoznanie się z tym jak wygląda praca na każdym z nich. Analiza porównawcza zebranych danych pozwoliła wyłonić wydajniejsze rozwiązanie, którym okazał się silnik Unity.

Słowa kluczowe: wirtualne muzeum; modele 3D; silniki gier; analiza porównawcza

*Corresponding author

Email addresses: **agata.ciekanowska@pollub.edu.pl** (A. M. Ciekanowska), **adam.kiszczak-glinski@pollub.edu.pl** (A. K. Kiszczak – Gliński)

1. Introduction

The number of internet connections across the globe is growing rapidly. Thanks to this process, developing virtual entertainment and education systems is easier and faster than ever. This trend also includes virtual museums, which are quickly gaining in popularity. Considering recent restrictions related to the pandemic, the value of virtual exhibitions is even greater lately.

In "The virtual museum", F. Antinucci attempted to create a definition of virtual museum [1]. According to the author, virtual exhibition is not a real museum, as watching 3D models cannot replace the impression of watching an exhibit in real life. The author underlines the importance of experiencing a real museum space. Such visit makes perceiving sizes, volumes and textures of artefacts easier. That said, he values virtual museums, for the flexibility to create exhibitions which are difficult or even impossible to create in real life. In the discussed work, virtual museums are considered by the author more as a supplement to the existing, real life exhibitions rather than a separate entity.

J. Derwisz in "Współczesne technologie multimedialne w wirtualnej rekonstrukcji oraz prezentacji historycznych obiektów architektonicznych" indicates issues with the reconstruction of objects that changed their form or appearance over the years [2]. Further, she points out, that such exhibitions often do not have a sufficient financial support. The author suggests 3D scanning and modelling of the artefacts as a viable solution for those complications.

Possible methods of scanning objects in 3D are presented in „An approach to computer-aided reconstruction of museum exhibits" by J. Kęsik, J. Montusiewicz and R. Kayumov [3]. One of the given techniques is to use a stationary device where an object can be placed and scanned. A disadvantage of such technique is the risk of damaging the exhibit during the transfer process. On the other hand, there is a safer method available – a handheld device. Using a mobile device minimizes the risk of a destruction of the artefact.

In the „Comparison of Unity and Unreal Engine" by A. Šmíd the author examines title game engines using a game created especially for this purpose [4]. The game was deployed in both of the engines, and later examined on a desktop computer, laptop, Android smartphone and Virtual Reality system. The author thinks that Unity was easier game engine to learn, the compilation time was shorter and the final size of the project was smaller. Alternatively, the Unreal Engine editor allowed creating scripts in visual Blueprint system, and had more advanced tools to create materials or generate terrain and vegetation. The final conclusion was that Unity is better solution when creating simpler projects on mobile platforms, and Unreal Engine is better suited for develop-

ment of more complex games on desktop computers or consoles.

A study of optimization methods in Unity game engine is included in the „Metody optymalizacji wydajności silnika Unity 3D w oparciu o grę z widokiem perspektywy trzeciej osoby" by K. Siarkowski, P. Sprawka and M. Plechawska-Wójcik [5]. For the research, the application with the third-person perspective was created. Parameters affecting performance such as occlusion culling, clip planes, batching and lighting were examined. Appropriate changes to the mentioned parameters gave positive results and positively affected the game performance.

E. Puławski and M. Tokarski within „Wykorzystanie postprocesingu i jego wpływu na wydajność renderowania w silniku Unreal Engine 4" article examined the Unreal Engine performance based on parameters such as lens flare, antialiasing, bloom or depth of field [6]. Researchers used an application created specifically for this purpose. The outcome showed that antialiasing is the most aggravating effect and bloom – the least.

2. Selected game engines overview

Game engines chosen as the research subjects in this work are Unity and Unreal Engine. Unity is an engine developed by Unity Technologies and Unreal Engine was created by Epic Games [7][8]. Both of them allow users to create 2D or 3D cross-platform applications such as games, visualisations, product configurators, interactive exhibitions or even virtual museums. According to the survey made in 2020 by Unity Technologies, Unity was the most popular software used to create games on mobile devices [9]. A diagram showing survey results can be seen on Figure 1.

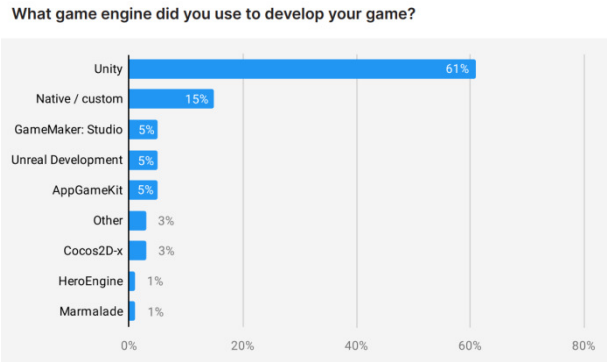

Figure 1: Game engines popularity in 2020 in creating mobile apps [9].

Based on the Figure 1, the Unity engine was selected to develop mobile apps 61% of time. Unreal Engine ranks as the second most popular solution available to general use, along with GameMaker: Studio and AppGameKit (each of them scored 5%).

Another research was made in 2019 and used a special script [10]. The script investigated games published on Steam platform for the usage of different game engines. Collected data is valuable but unfortunately it's not the most precise. To be involved in the research, the game must have had a Wikipedia page where an information about used game engine had to be filled. The results can be seen on Figure 2.

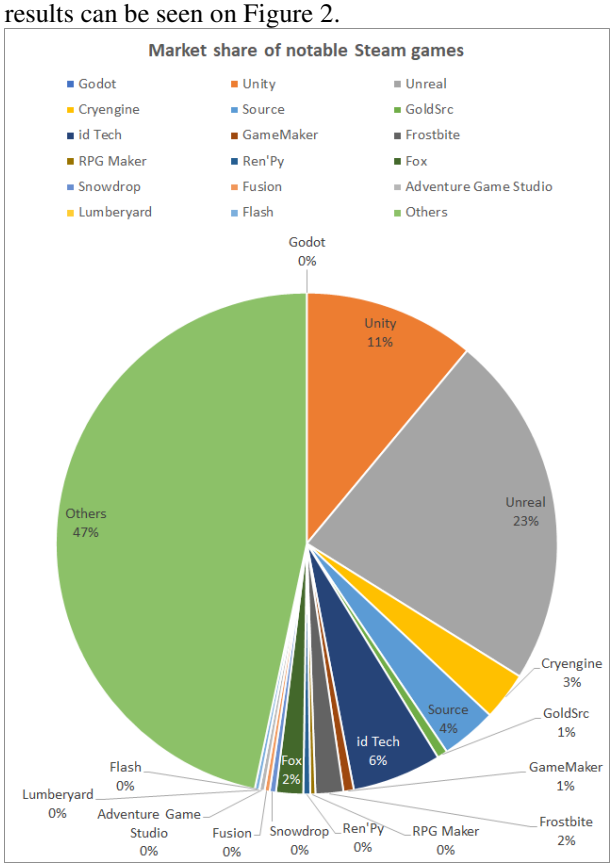

Figure 2: Market share of notable Steam games in 2019 [10].

According to this research (as can be seen on Figure 2), the most popular game engines were Unreal Engine which scored 23% and Unity with the score of 11%. The popularity aspect had a great importance in the choice of game engines to be used in our research.

It is worth adding that both of the examined engines at the moment of writing this paper had a policy which allowed free use for the non-commercial purposes. Moreover, Unity and Unreal have extensive documentation, which make application development easier and faster.

3. 3D scanned models

Artefacts used in this research originate from the Silk Road in Central Asia. The 3D models of these exhibits can be seen on Figure 3.

Figure 3: 3D models of exhibits from the Silk Road.

From the top left corner, clockwise – a lamp fragment, a camel-shaped jug, a beverage bottle and a perfume vessel.

The four scanned and textured artefacts presented on Figure 3 were used creating the virtual museum. Each model has a few variations with mesh simplified in a different degree. A 3D mesh is the structural build of a 3D model. 3D meshes use reference points in X, Y and Z axes to define shapes. Availability of multiple models in several quality levels allowed achieving a satisfying performance.

3.1. Obtaining the models

The models shown in the Figure 3 were created using Artec Space Spider handheld scanner, made by Artec 3D. The scanner is shown on the Figure 4 [11].

Figure 4: Artec Space Spider scanner [11].

The Space Spider has an ability to capture objects up to 2000 cm^3. Its maximum scanning capability is 1 million points per second with the 7.5 FPS frame rate. Moreover, it enables obtaining textures of scanned pieces with a 1.3 Mpx resolution [12].

Finished scans of the four exhibits were later processed using AutoCAD and Artec Studio software.

It is also possible to use stationary devices to scan exhibits but it poses a risk of damaging or even destroying object while moving it [13].

4. The virtual museum

The virtual museum was deployed in both Unity and Unreal Engine. Efforts were made to make the two scenes as similar to each other as possible, e.g. using the same textures, materials and 3D model of a museum or placing artefacts in an identical arrangement, etc. Alike settings for the museum were applied as well.

Figure 5: Virtual museum implemented in the Unity game engine.

Figure 6: Virtual museum implemented in the Unreal game engine.

The model of the museum area including lamps, sconces, pedestals and a front door was created using Blender software which is free to use [14]. A virtual visitor can move around in the created space from the first-person perspective. The user also can interact with each exhibit by pressing ‘E’ key on a keyboard whilst standing nearby the chosen artefact. An information about this function is displayed when appropriate for convenience.

Figure 7: Exhibit view made in the Unity game engine, allowing interaction with the model.

Figure 8: Exhibit view made in the Unreal game engine, allowing interaction with the model.

The purpose of the view presented in Figures 7 and 8, is that the person visiting the museum can rotate the chosen object and read information about it. An informative note about the object is presented at the bottom of the screen. This gives the user a chance to take a better look at every exhibit and to discover its origins.

Scripts responsible for logic of the projects were created in C# and the Blueprint system for Unity game engine and Unreal Engine respectively.

The Level Of Detail (LOD) technique was used for optimization of performance on available hardware. The LOD is a tool allowing for improvement of the performance of a scene by loading a lower quality meshes in situations where user wouldn't notice the lack of details in models. It is based on the distance between the object and a camera (Unity) or percentage of screen surface occupied by model (Unreal) [15][16]. Considering very high complexity of models in the best quality (from 508028 up to 3199707 of mesh triangles), adding the LOD was necessary to ensure smooth and correct performance of the application. The highest quality model of an artefact is only displayed when the user is standing as close as possible to the object or enables the view presented in Figures 7 and 8.

5. The experiment

The experiment was conducted on two PCs and two laptops – each of them with different hardware specifications, listed in the Table 1.

Table 1: Specifications of computers used in experiment

Computer	Processor (CPU)	Graphic card (GPU)	RAM
PC 1	AMD Ryzen 7 3700x, 3.60GHz	AMD Radeon RX580	32 GB
PC 2	Intel i5-4460, 3.20GHz	AMD Radeon R9 270x	8 GB
Laptop 1	Intel i5-8250U, 1.60GHz	NVIDIA GeForce 940MX	8 GB
Laptop 2	Intel i5-7200U, 2.50GHz	NVIDIA GeForce 940MX	8 GB

In order to carry out the comparative analysis and to verify the efficiency of both game engines, six main criteria for examination were established:

- CPU load (measured as percentage),
- GPU load (measured as percentage),
- RAM load (measured as MiB),
- FPS value - Frames Per Second,
- subjective authors opinion on working with examined game engines and intuitive of their user interfaces (measured on 0 to 5 scale),
- final project size (measured as MB).

The experiment duration was 1 minute and it was repeated 10 times on every computer for each game engine. Performance data was gathered every second. To make tests comparable, all runs involved a similar course and the same order of exhibit examination. An individual artefact was observed in a view presented in Figures 7 and 8 for about 15 seconds, and after that, a virtual visitor proceeded to another exhibit.

Data describing CPU and RAM usage was collected using Performance Monitor built in the Windows 10 operating system. It allows saving hardware usage information limited to only one process, which was a crucial functionality for the experiment. Open Hardware Monitor software was used to gather information about GPU load and FPS values were captured with the help of Fraps programme. What is important, FPS value was limited to 60 in both instances. The reason behind that is this FPS value is considered optimal. Images projected in this frame rate are smooth, and GPU load is often reduced if the graphic card is powerful enough.

5.1. Results

Data shown on every diagram presented in this subsection is an average based on averages coming from data collected in each experiment series and is used for verification of the performance of both examined engines.

Figure 9 presents a diagram containing average processor usage data.

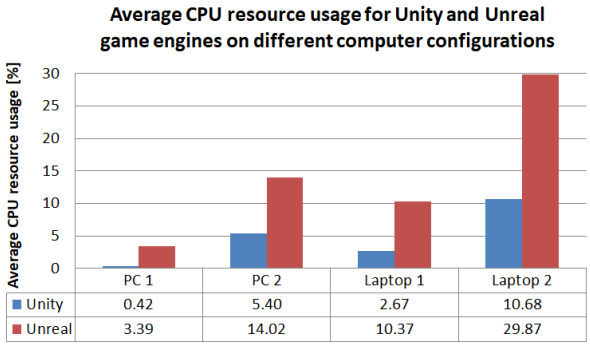

Figure 9: Average CPU usage.

The best performance for both Unity and Unreal engines was carried out by Laptop 1 in the laptops group and PC 1 in desktop computers group. An interesting outcome from this graph is the fact that PC 2 configuration has shown a worst performance than Laptop 1, despite having slightly better hardware specification. After delving into more specific information about each of the examined processors, possible explanation of this phenomenon was discovered. It is possible, that the number of threads in each unit was a decisive factor. CPU inside the PC 1 has 8 cores and 16 threads and the PC 2 has only 4 cores and 4 threads available. The processor in Laptop 1 has 4 cores alike PC 2, but 8 threads. Laptop 2 has only 2 cores and 4 threads. Quantity of cores can make a difference too, which is clear, when comparing PC 2 and Laptop 2.

Figure 10 represents collected information about average graphic card usage.

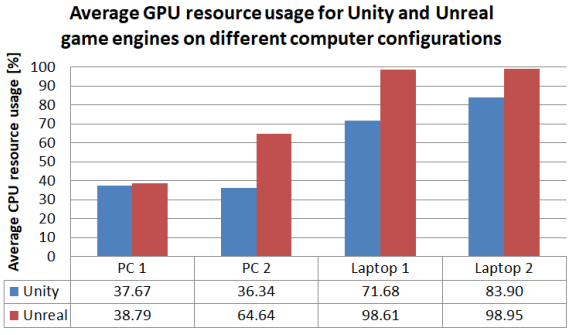

Figure 10: Average GPU usage

In most of the cases the Unity game engine achieved a better result compared to its competitor. The most significant difference between the two can be observed on the PC 2 – Unreal is nearly 2 times less efficient. For both of the laptops, the Unreal engine achieved similar result which can be rounded to approx. 99% of GPU usage. Despite the fact that both laptops have the same graphic card, the Unity engine had slightly better performance in this category on Laptop 1.

Figure 11 presents an average RAM consumption for each of the examined game engines.

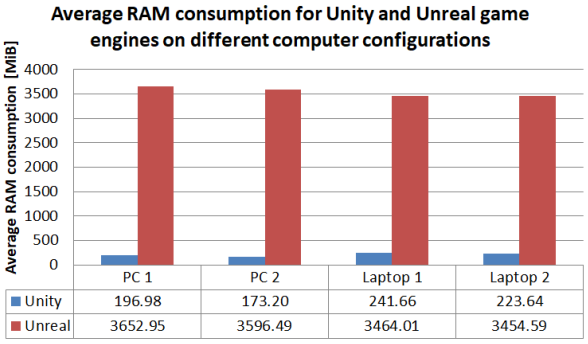

Figure 11: Average RAM consumption.

The difference between the two subjects of study is significant. Whilst testing Unity, RAM usage value has never exceeded 250MiB. For the Unreal game engine, the value fluctuated around 3500MiB.

Diagram of an average frames per second value is shown on the Figure 12. On the PC 1 both of the examined engines achieved nearly identical score which is very high. Most of the time, the scene on PC 1 was running at approx. 60FPS and that means it was stable and smooth in both examined cases. PC 2 achieved very similar score when it comes to the Unity engine. When running Unreal Engine, PC 2 performed slightly worse, but still excellent. In regard to laptops, Unity was capable of generating more FPS than Unreal by about 10 on both Laptop 1 and Laptop 2. The worst score for Unreal game engine was registered on Laptop 2 and it was 40 frames per second. It is still satisfactory result, but in this case, the quality was about ⅓ worse compared to PC 1 and PC 2.

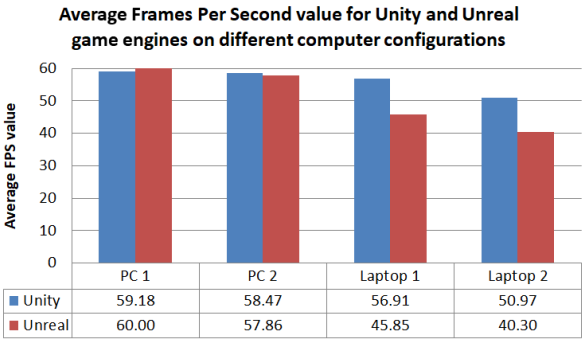

Figure 12: Average FPS value.

It is worth to compare average FPS diagram with average GPU usage graph. Unreal game engine in laptops group used about 99% of graphic card resources. That means generating more FPS than what can be seen on Figure 12 is not possible without further optimization. Alternatively, Unity did not use up all of the GPU computing power. That means improving achieved FPS values is still possible.

Figure 13 shows sizes of the final built projects. The size of the project made in Unity is about 50% smaller compared to the Unreal Engine.

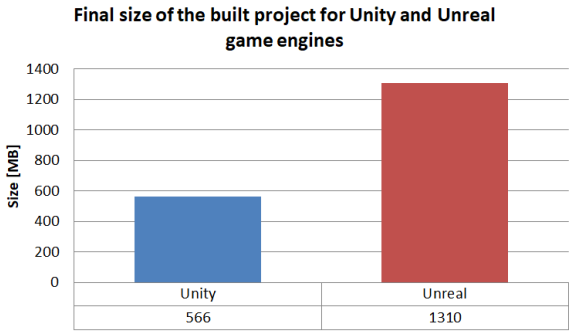

Figure 13: Final size of the built project.

The intuitiveness of engines User Interface and opinion on working with examined game engines are subjects to subjective evaluation. That is why there's no diagram with comparison results included in this category. In the opinion of the authors, both interfaces are similar. They allow the user to personalize the editor by changing positions of the individual windows in the workspace. One of the Unreal Engine biggest advantages is the possibility of programming scripts in C++ as well as in the Blueprint. The Blueprint is a visual programming system which allows users with no experience in programming as well as professionals to write full-fledged scripts. It is worth adding that Unity allows writing code in C# which is easier to learn than C++ but the user has more capabilities available using the Epic Games product. From the hardware perspective, the Unreal editor is more resource-demanding compared to Unity. Despite the fact that both of the examined engines are free to use up to a certain revenue from sales, the Unreal Engine is targeted more towards bigger game developing studios, whilst Unity engine is addressed to independent developers.

In order to compare values measured in different units (e.g. RAM and FPS), a grade is assigned to the selected criterion. The grade depends on the average score obtained during the experiment. All configurations were taken into consideration. Ratings for the ranges of results are listed in the Table 2.

Table 2: Rates for the ranges of results

Rating	CPU [%]	GPU [%]	RAM [MiB]	FPS	Size [MB]
5	0-5	0-30	-	60-55.01	-
4	5.01-10	30.01-40	-	55-50.01	-
3	10.01-15	40.01-50	-	50-45.01	-
2	15.01-20	50.01-70	500-	45-40.01	600-
1	20.01-25	70.01-90	500+	40-35.01	600+
0	25.01+	90.01+	-	35-	-

An average score for Unity engine on all of the computer specifications in the aspect of CPU usage can be rounded to 4.79% and for Unreal Engine – 14.41%. That means according to the Table 2 Unity's rating is 5 and Unreal's rating is 3.

In the GPU usage category an average result for Unity after rounding was 60.23, resulting in a grade 2. After calculations, Unreal result was 81.63, so it was rated 1.

The RAM consumption evaluation slightly varies-from previous criteria. Considering the differences in amount of RAM available in PC 1, and the fact that there is a big difference between the two compared engines in this case, there are only two grades available. The threshold was established based on the corresponding diagram. For Unity the grade is 2, and Unreal received 1.

An average score for FPS values for both of the game engines was also calculated. In this category Unity achieved an average of 56.38 FPS which equals rating 5. Unreal Engine accomplished a similar average value of FPS, which was 51, resulting in grade 4.

Due to the fact that interface intuitiveness and the simplicity of use of both of the examined engines is a subjective value, there are no ranges for the rating in this category. Based on the described advantages and disadvantages in the opinion of authors, both engines score 5 in the 0 to 5 scale.

In the final category, project size, there are only 2 grades available. As the size of project made in Unity was below 600MB, it receives rating of 2 and given the fact that the exhibition size made in Unreal Engine was above 600MB, its rating is 1.

Achieved results and given grades are presented in Table 3.

Table 3: Results and ratings given to the examined game engines

Criterion	Unity		Unreal	
	Score	Rating	Score	Rating
CPU load	4.79%	5	14.41%	3
GPU load	60.23%	2	81.63%	1
RAM load	<500 MiB	2	>500 MiB	1
FPS value	56.38	5	51	4
Overall experience	-	5	-	5
Final project size	<600 MB	2	>600 MB	1

Based on all of the information contained in this subsection, the resulting Table allowing comparative analysis was created. Listed criteria are the same as the ones mentioned in chapter 5. Weights of all of the six criteria add up to the number 1 and – more importantly – the higher the weight, the criterion had more importance for the whole project and the final rating. Unity and Unreal columns are meant for ratings of these game engines, bearing in mind the specific categories. The evaluation for a specific category and engine is derived from multiplying weight and rating from Table 3.

Table 4: Resulting table

No.	Criterion	Weight	Unity	Unreal
1	CPU load	0.1	0.5	0.3
2	GPU load	0.3	0.6	0.3
3	RAM load	0.2	0.4	0.2
4	FPS value	0.25	1.25	1
5	Overall experience	0.1	0.5	0.5
6	Final project size	0.05	0.1	0.05
	Totals:	1	3.35	2.35

As is seen in the Table 4, Unity engine has better final score (3.35) than Unreal Engine (2.35) meaning that Unity is more efficient tool for creating virtual exhibitions of 3D scanned models.

6. Conclusions

The experiment created for this work purposes has shown that the virtual museum can be made in Unreal and Unity game engines. However, achieved effects differ slightly. While both editors are similar in use from the creator's point of view and the visual side of created exhibitions is similar too, the final products aren't identical in terms of performance. In the most cases Unity has advantage over the rival engine. The biggest difference between the two, to the disadvantage of the Unreal Engine can be seen in the RAM usage comparison diagram and final size of the built project. For Unreal engine it was 3542 MiB and 1310 MB accordingly and for Unity – 209 MiB and 566 MB. Unity also wins in the CPU resource consumption – it used half as much resources as the competitor (appropriately 4.8% and 14.4%). Results in the GPU resource consumption are in favour of the Unity engine as well, which resource usage oscillated around 57%, being 18 percentage points lower than second game engine under consideration. When it comes to the amount of generated FPS, similar results can be seen on desktop computers – both Unity and Unreal accomplished an average of 59 frames per second, but on both of the laptops Unreal Engine performs worse - its mean was 43 FPS which was a score worse than Unity by 10 frames.

Final grades for Unity and Unreal Engine were 3.35 and 2.35 accordingly. These ratings beg the question - why is Unreal Engine still so popular? From the subjective point of view the created scene was looking better in Unreal with less work put into visual aspect of it. Blueprint system is definitely an advantage of this engine too. The engine also provides more advanced tools, resulting in greater popularity among professional users and game developing studios.

Considering high RAM usage compared to Unity, it is worth further examination. One of the possibilities is the potential difference between applications created using Blueprints and C++ and their RAM consumption.

References

[1] F. Antinucci, The virtual museum, Archeologia e Calcolatori, supplemento, 1 (2007) 79-86, http://www.archcalc.cnr.it/indice/Suppl_1/6_Antinucci.pdf [02.07.2021].

[2] J. Derwisz, Współczesne technologie multimedialne w wirtualnej rekonstrukcji oraz prezentacji historycznych obiektów architektonicznych, Wiadomości Konserwatorskie, 34 (2013) 82-91, http://yadda.icm.edu.pl/yadda/element/bwmeta1.element.baztech-909a5d65-29b1-46fd-bfa5-3c76fb7efac1/c/Derwisz.pdf [02.07.2021].

[3] J. Kęsik, J. Montusiewicz, R. Kayumov, An approach to computer-aided reconstruction of museum exhibits, Advances in Science and Technology. Research Journal, 11 (2017) 87-94, http://yadda.icm.edu.pl/yadda/element/bwmeta1.element.baztech-73ba2929-b9ff-4741-978b-bf780441dd3f/c/kesik.pdf [02.07.2021].

[4] A. Šmíd, Comparison of Unity and Unreal Engine, Bachelor Thesis, Czech Technical University, Prague, 2017, https://dcgi.fel.cvut.cz/theses/2017/smidanto [02.07.2021].

[5] K. Siarkowski, P. Sprawka, M. Plechawska-Wójcik, Metody optymalizacji wydajności silnika Unity 3D w oparciu o grę z widokiem perspektywy trzeciej osoby, Journal of Computer Sciences Institute, 3 (2017) 46-53, https://doi.org/10.35784/jcsi.592 [02.07.2021].

[6] E. Puławski, M. Tokarski, Wykorzystanie postprocesingu i jego wpływu na wydajność renderowania w silniku Unreal Engine 4, Journal of Computer Sciences Institute, 10 (2019) 54-61, https://doi.org/10.35784/jcsi.206 [02.07.2021].

[7] J. Haas, A History of the Unity Game Engine, An Interactive Qualifying Project, Worcester Polytechnic Institute, Worcester, 2014, https://digital.wpi.edu/pdfviewer/2f75r821k [02.07.2021].

[8] A. Szewczyk, Oczekiwania fanów elektronicznej rozrywki wobec grafiki w grach komputerowych, Studia Informatica Pomerania, 40 (2016) 71-85, http://wneiz.pl/nauka_wneiz/studia_inf/40-2016/si-40-71.pdf [02.07.2021].

[9] 2021 Gaming Report - Unity insights from 2020 and predicted trends for 2021, https://images.response.unity3d.com/Web/Unity/%7B4645ad28-63a3-4348-bc5b-dc09f2811419%7D_Unity_2021-Gaming-Report.pdf [02.07.2021].

[10] Research of the market share of game engines on Steam, from over 60,000 Steam games, https://www.reddit.com/r/gamedev/comments/8s20qp/i_researched_the_market_share_of_game_engines_on/ [02.07.2021].

[11] Artec Space Spider, https://skanery3d.eu/skanery-3d/artec-spider/ [02.07.2021].

[12] Artec 3D Technical specifications, https://www.artec3d.com/portable-3d-scanners/artec-spider-v2#specifications [02.07.2021].

[13] M. J. Wachowiak, B. V. Karas, 3D Scanning and Replication for Museum and Cultural Heritage Applications, Journal of the American Institute for Conservation, 48 (2019) 54-61, https://www.si.edu/content/MCIImagingStudio/papers/scanning_paper.pdf [02.07.2021].

[14] A. Salwierz, T. Szymczyk, Metody wytwarzania realistycznych pomieszczeń – skanowanie 3D oraz modelowanie 3D, Journal of Computer Sciences Institute, 14 (2020) 101-108, https://doi.org/10.35784/jcsi.1584 [02.07.2021].

[15] Unity - Level of Detail (LOD) for meshes, https://docs.unity3d.com/Manual/LevelOfDetail.html [02.07.2021].

[16] Unreal Engine - Creating and Using LODs, https://docs.unrealengine.com/4.26/en-US/WorkingWithContent/Types/StaticMeshes/HowTo/LODs/ [02.07.2021].