Game Engine Architecture and Comparative Study of Different Game Engines

Chaitya Vohera
Dept. of Information and Technology
Thadomal Shahani Engineering College
Mumbai, India
chaitya321@gmail.com

Heet Chheda
Dept. of Information and Technology
Thadomal Shahani Engineering College
Mumbai, India
heetc27@gmail.com

Dhruveel Chouhan
Dept. of Information and Technology
Thadomal Shahani Engineering College
Mumbai, India
dhruveel.chouhan.3@gmail.com

Ayush Desai
Dept. of Information and Technology
Thadomal Shahani Engineering College
Mumbai, India
desai.ayushh@gmail.com

Vijal Jain
Dept. of Information and Technology
Thadomal Shahani Engineering College
Mumbai, India
vijal.jain@thadomal.org

2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT) | 978-1-7281-8595-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICCCNT51525.2021.9579618

Abstract— **Nowadays Game Engines have become an integral component of the game development environment. Not only do they accelerate the game development process but also facilitate the integration of gaming modules like animations, graphics, artificial intelligence, and physics using their in-built functionalities. Game Engines also provide a major advantage of reusability of their components making them highly scalable and modifiable. Game engines can be used by the developers to construct and develop games for consoles and different types of platforms such as Android, IOS, Desktop, and many more. This paper eludes the game engine architecture and its constituents and illustrates the features and comparative analysis between four popular game engines namely, Unity, GameMaker, Unreal, and CryEngine. The parameters of this comparison are based on the game engines' technical and non-technical aspects. In conclusion, users will be assisted by the extensive overview provided by this paper in choosing the most preferable engine for their game according to the requirements.**

Keywords—Cry Engine, **Game Engine Architecture, GameMaker, Unity, Unreal**

I. Introduction

Every day, different types of games are created whether it is a desktop game, a console game, or a mobile game. People associated with the game design and development come from various backgrounds [1]. Due to this, the people not familiar with any prior coding knowledge or programming languages such as C++, C# or Java, may find it difficult to create a game and hence everyone cannot attainable it. This is when the Game Engines come into the picture. Game Engines help to design and develop a game with the help of templates and assets that can be reused in further applications, and hence making it not necessary to have a profound experience in programming. With the help of game engines, a single game developed can be deployed to various platforms such as PC, consoles, or mobiles. Nowadays, the importance of gaming has increased to such an extent that its events are being organized in the Olympics. Although the majority of people know about different games, they are hardly aware of the architecture in which a particular game is created. This architecture is called the Game Engine. For lone developers or teams who only want to focus on developing the greatest game possible, game engines deliver a greater advantage over starting from scratch. The formatter will need to create these components, incorporating the applicable criteria that follow.

The name "game engine" should probably be reserved for software that is adaptable and may be used as the foundation for a variety of games without major changes. The majority of the game engines are meticulously built and tailor-made to run a specific game on a specific hardware platform. Even the most diversified and versatile game engines which are compatible with multiple platforms, actually are most efficient for games in one specific genre, such as shooting games, racing, or sports games [1]. Thus, it is safe to say that the more the generic nature of a game engine, the less is its efficiency to run a specific game on a specific platform.

Game engines act as a base for implementing game-related tasks like exhibiting graphics, interpreting data, physics-related computation, memory management and to expedite game developers for concentrating on the precisions which help to make the game unique. Game engines play an integral role in the integrated game development environment. They consist of congenital visual development tools and software components that can be used again. Game engines aim at reincarnating the real-world elements into the digital world by the alliance of graphics, audio, animation, networking, physics, and AI [2]. With the concept of video gaming on the decline, people prefer endowing time in real-time cloud gaming and mobile games, which are adaptable to various platforms and operating systems. Gaming engines ensure this consistency of gaming experience by providing extensive graphics and quality and the conventional joystick pattern.

This paper explains what a game engine means and what are major components of it. It exhibits the concept of a game

engine, its architecture, and the surveyed information about the four most famous game engines used today are briefly studied and a comparison to offer the criteria that must be considered before choosing a game engine for developing games has been portrayed. The reason to select these four-game engines is that they are open-source, easy to start with, and are freely available to the users.

II. Game Engine Architecture

The study is adopted from [1,3,4,14,15] which explains the different game engines. Based on the study, a generic game engine framework is proposed in this paper. It also outlines the various components involved in the construction of the game engine.

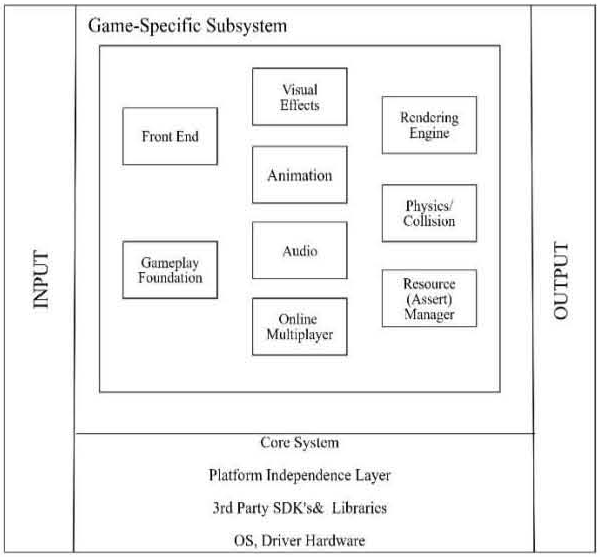

Fig 1: Generalized Game Engine Architecture

1) *Operating System:* On a computer, the operating system (OS) is always active. On a single computer, it coordinates the execution of numerous programs, one of which is the game. The operating system on a console is frequently merely a thin library layer. In most cases, the game "owns" the entire machine.
2) *Device Drivers:* A is software that manages hardware devices in order for them to connect with the computer's operating system is called a device driver. The computer will be unable to send or receive data from hardware devices, and the hardware device will fail to execute a certain duty or function without device drivers. Hardware Driver is an alternative name for device drivers.
3) *Target Hardware:* Different gaming platforms include iOS and Android phones, Microsoft Windows and Linux PCs, Xbox, PlayStation, Nintendo, and many more. The job of the Target Hardware component is to indicate the platform on which the game is going to be played
4) *Third Party SDK and Libraries:* To build a reusable component, third-party software is created by a body other than the original seller of the development platform. These libraries provide developers a one-of-a-kind opportunity to incorporate pre-tested, reusable software, reducing development time and cost. This enables the developer to concentrate on the game's main aspects that are important to players.
5) *Platform Independence Layer (PIL):* Most gaming engines must be able to run on several hardware platforms. As a result, most gaming engines include a platform independence layer, which sits on top of the hardware, drivers, operating system, and other third-party software and protects the remainder of the engine from the majority of platform information. PIL guarantees that all hardware platforms behave invariably. Because there is so much diversity between platforms, this is important.
6) *Core System:* Core systems can be called the toolbox of a game engine. A game engine needs different kinds of tools and software which are basically core systems. Here are a few instances of the services often provided by the core layer. Memory management, math libraries, string hashing, and asynchronous file I/O are just a few of the features available.
7) *Resource Manager:* It is a uniform interface (or combination of interfaces) for accessing all forms of game assets (fonts, materials, objects, and so on) as well as other engine input data.
8) *Rendering Engine:* Rendering is in charge of showing visuals on the screen, which might be in 2D or 3D. It is usually the most complex subsystem. At this level, the goal of the rendering engine is to show or synthesize basic shapes as soon as possible, with little care for which parts of a picture are visible.
9) *Visual Effects:* Visual effects improve the quality of the game. They make the game lively. Particle systems, dynamic shadows, light mapping are few examples of different visual effects supported by game engines.
10) *Front End:* For a variety of reasons, most games use 2D visuals placed over a 3D environment. These include the graphical user interface (GUI), in-game menus, and the head-up display (HUD). On-Game Cinematics or Full Motion Video can also be used in the front end.
11) *Physics/Collision Engine:* Aids in simulating physics in the Project to guarantee that objects accelerate and behave appropriately to collisions, gravity, and other factors. If Collisions is not used, objects would not collide with each other and maybe they will overlap or penetrate each other and make it impossible to interact with other game objects or make logical sense of the game. The motion (kinematics) of rigid bodies and the forces and torques (dynamics) that generate this motion are generally the sole things that rigid body dynamics are concerned with. Collision and physics are frequently inextricably linked. This is because collisions are nearly always addressed as part of the physics integration and constraint fulfillment logic when they are recognized.
12) *Animation:* Every game, whether 2D or 3D, necessitates the use of animations. Sprites, which are merely a collection of pictures shown over time, are used to create 2D animations. Skeletal animation allows an animator to position a complex 3D character model using a basic bone system. The 3D mesh's vertices move in synchronization with the bones.
13) *Audio:* In every gaming engine, audio is equally as vital as visuals. It creates in-game background music and

noises that are triggered by certain events like the sound of footsteps while walking or gunshot while shooting.

14) *Online Multiplayer:* Networking is a crucial component for implementing multiplayer capabilities. There are several techniques to programming multiplayer games, such as single-screen multiplayer, split-screen multiplayer, networked multiplayer, and massively multiplayer online games (MMOG).
15) *Gameplay Foundation Systems:* The game's action, the rules that govern the virtual reality wherein the game is being played, the abilities of the player character(s) and other characters and things in the environment, and the player's aims and objectives (s) all come under the term gameplay. It also has a Scripting System, which game engines utilize to create above mentioned different gameplay components easier and faster.
16) *Game-Specific Subsystems:* Game developers and designers collaborate to develop features of the game above and beyond all other engine components. Player character dynamics, multiple camera systems, non-player characters (NPCs) controlled by AI, weaponry, other objects like vehicles, and so on are examples of Subsystems.

III. Literature Review

The future of next-generation gaming engines appears to be bright and exciting for many creators. As a result, the game engine will be able to create a considerably more intricate game with high-quality detailing. The following game engines are explored because they make understanding and developing games for various platforms easier.

A. Unity Review

Unity is an IDE that allows users to create video games, 3D animations, and design modelling. The IDE can be run on Windows as well as on macOS. But still, it can design games for several platforms such as Desktop (Windows macOS, Linux), whereas when it comes to mobile devices it can develop games for Windows Phone, iOS, Android, and different consoles. The user just needs selected platforms Software Development Kit (SDK) for Unity to create the application on the relevant platforms[5]. Unity Engine was created in C++ and C# and the scripts support C#, JavaScript, and Boo code[6]. Scripts play an important role in assigning the Game Object's animation or real-time transition. Unity incorporates a variety of advanced components such as Nvidia's physics engine known as PhysX, an animation system by Mecanim, a built-in terrain editor, MLAPI for multiplayer networking. It also uses the MonoDevelop (a code editor), which immediately compiles the code automatically and injects it into the game after even a small modification in the MonoDevelop code. The compilation is done using JavaScript and C# compilers present in Unity. The Unity console window displays compilation problems[16]. The Unity3D GUI makes it easier for a novice developer to approach, write, and design the transition of the Game-Object. Unity is ideal for indie developers, small studios, and those of us who have always desired to create our games. Its enormous user base and incredibly active user community enable everyone from newcomers to experienced professionals to rapidly receive answers and exchange knowledge[6]. It also allows designers to shop for assets online. Designers might also create their elements on their own. Unity supports assets created in Blender, 3ds Max, Cinema 4D, Softimage Adobe Photoshop, Maya, and many more [5]. Famous Unity games include Call of Duty: Mobile, Temple Run Trilogy, Assassin's Creed: Identity, and Escape Plan.

B. Unreal Review

The Unreal Engine (UE) was designed by Epic Games, which is a game development software. The first version was launched in 1998, and the most recent version is 4.20, which was released in 2018[8]. The engine was created using C++ and it features a high degree of portability, supporting a wide range of desktop, mobile, console and virtual reality platforms. The following are the most significant advantages of using a game engine like Unreal Engine: reusable code using libraries, the concept of object-oriented programming, and the technology of machine portraits. The engine includes dedicated libraries for game creation; it's not about figuring out how to make a game; instead, it's about highlighting the significance rather than the technical aspects. In addition, using object-oriented programming, makes it viable to create magnificence the traditional manner, this magnificence includes all of the vital code to create an individual on the spot. Finally, the developer uses complex algorithms to calculate the portraits the sport needs. Unreal Engine makes use of the Disney's Physical Based Rendering [9]. Constraints with hardware and frames per second were the first and major difference between movies and videogames. It's critical to optimize the environment so that calculations performed in real time can exceed 30 frames per second. To overcome the problem, Unreal Engine provides a complicated structure that allows for fast frame rates and excellent fidelity. Using Level of Detail (LOD) simplifies the 3-D fashions relying at the viewing distance of the camera, decreasing the quantity of polygons rendered at the screen. Using Unreal Engine's gear made it possible to grow the visible best of the video game's environment while not having any overall performance issues [9]. Remarkable games that have been created using Unreal Engine are Fornite, Street Fighter V, Borderlands 3, Dragon Ball Fighter Z, Ark: Survival Evolved.

C. GameMaker Review

Initially released under the name Animo in 1991 by Mark Overmars and later Game Maker until 2011, GameMaker is a game engine developed by YOYO Games used to create 2-Dimensional (2D) cross-platform games. It is written in C++ and C#. GameMaker is suitable for a beginner since no programming skills or prior knowledge is required, making it very easy to use. It enables users to develop games with a single piece of code and publish them to run on numerous platforms such as Android, iOS, PlayStation, Xbox, Windows Desktop, OS X, HTML5, Windows UWP, and Nintendo Switch. [12]. Apart from creating 2D games, it has a special feature that allows a user to add 3D graphics and physics [10]. However, the functionality of 3D games is restricted [12]. GameMaker offers recommendations to increase the readiness with which the user engages with key programming concepts and practices when using this visual programming software [11]. The latest available edition of GameMaker available in the market is GameMaker 2, first released in the year 2017. GameMaker offers various product licenses ranging from $39 to $799 depending upon the user choice. Some of the famous games created in GameMaker are Undertale, Hotline Miami,

Shovel Knight: Treasure Love, Hyper Light Drifter, Katana Zero.

D. CryEngine Review

Cryengine is the gaming engine designed by Crytek. Cryengine 1 started off with the development of the first player shooting game, FarCry and its sequels. The scripting of the gaming engine is done in C++ and Lua programming languages [7]. It allows game development for multiple platforms including XBox One, Playstation 4, Windows, Linux, PSVR, Oculus Rift.The engine handles real-time asset conversion and optimization, allowing for cross-platform modifications to all aspects of the game development process. This improves development speed and quality while considerably lowering the risk of developing multiplatform games. Using CryEngine gives access to both source codes and the gaming engine, therefore there is more facilitation of flexibility and customization. Cryengine is well known for creating extremely good quality graphics and game performance. Ryse: Son of Rome by CryEngine was awarded the SIGGRAPH award for being the best real time graphics game in the year 2014 when it was released [7]. Cryengine games are primarily based on the genre of first player shooting games. Some of the famous Cryengine games include Crysis and its sequels, Prey, Kingdom Come, Ryse: Son of Rome.

Previous studies reviewed did not include a detailed explanation of game engine architecture as well as a comparison table. Architecture was presented in those publications in a way that would have been challenging for a newcomer to understand. This study chose comparison parameters that will assist novice developers in selecting the optimal engine for themselves, whereas parameters in previous publications were more for advanced developers.

IV. Game Engine Comparison

The table 1 shows the comparison of the four most widely used game engines today. It describes their performances in terms of technical and non-technical aspect.

TABLE I
Comparison Of Game Engine

Sr No.	Parameters	Unity	GameMaker	Unreal	CryEngine
1.	Cross Platform	Consoles (Xbox, PlayStation, Wii U, Nintendo), OS or Desktop (macOS, Windows and Linux), Mobile devices (Android, Windows, iOS, Blackberry), WebGL	Consoles (Xbox, PlayStation, Nintendo), OS (Windows, macOS), Mobile devices (Android, iOS), HTML5	Consoles (Xbox, PlayStation, Switch), OS (Windows, macOS, Linux), Mobile devices (iOS, Android), HTML5	Consoles (Xbox, PlayStation, Oculus Rift), OS (Windows, Linux), Mobile devices(not supported)
2.	OS Support	Windows, Linux, Mac	Windows	Windows, Mac	Windows, Linux
3.	Programming Languages	C#, JavaScript, Boo	C#, C++	C++	C++, Lua
4.	Multifunctionality	2D, 3D	2D, 3D(FPS)	2D, 3D	3D
5.	Documentation	Best	Good	Good	Poor
6.	Difficulty Level (for Beginner)	Low	Moderate	High	High
7.	Artificial Intelligence	RAIN	Kynapse	Kynapse	Lua Driven AI
8.	Physics Engine	PhysX	Built-in	PhysX	Soft-Body
9.	Network/ Multiplayer	Supported	Supported	Supported	Supported
10.	Development Tools	Visual Studio, MonoDevelop	GameMaker Studio	BluePrint Editor, Visual Studio	FlowGraph, Visual Studio
11.	Terrain Design using Engine Tools	Medium	Most Difficult	High	Medium (Good in-built assets)
12.	Graphic Effects	Shadow effects, Particle system, Different types of lighting, Lens Flare	Particle system, Only 1 type of lightning	Shadow Effects, Particle system, Different types of lighting, Lens Flare	Shadow Effects, Particle System Different types of lighting, Lens Flare
13.	Libraries & Plugins	Maximum	Less than Unity and Unreal	Less than Unity	Least

14.	Technical and Community Support	Vast, active and supportive community	Small but active	Between Unity and CryEngine	Relatively small community
15.	Animations	Basic animation	2D Animation	SkeletalControl and supports dynamic animation (better than CryEngine)	Supports SkeletalControl and Facial Editor
16.	Modeling	Supports external modeling assets created in (Blender, 3ds Max, Maya, etc.) and it has built in asset store	Supports external editor like MMK Blender	Static Mesh Editor (not better when compared to CryEngine)	Provides Designer Tool, CryEngine's 3D object modeling tool.
17.	Pricing	Personal – Free Plus - $399/yr per seat Pro - $1800/yr per seat Enterprise - $2000/mo per 10 seats	30 days - free trial. Creator - $39 Developer (Desktop and Mobile) - $99 Console - $799 to $1500	Personal use - Free Professional Version - if revenue from the game is more than $3,000 /quarter then 5% of the game's gross income is charged or else free.	CryEngine is available for free including the full engine code. There is a 5% royalty fee levied after the first $5000 of revenue earnt.
18.	VR/AR	supports Oculus Rift, HTC Vive, Google DayDream, Cardboard, Gear VR, SteamVR,	Not Supported	supports Oculus Rift, HTC Vive, SteamVR, OSVR, Google VR/DayDream, Samsung Gear VR	supports HTC Vive, Oculus Rift

V. Discussion

1) *Unity* - For novices, the Unity engine is arguably the best option. Very well-written documentation, several courses, and ready-made templates allow you to rapidly learn all of the most significant engine features; there are several assets accessible. Unity is the only engine that includes a specific 2D mode. In comparison to the competition, animation, the production of video interludes, and the addition of audio effects are limited to basic functions. The Unity engine appears to be among the worst of the examined engines in terms of graphical quality. The engine is unsuitable for creating games with huge and complicated worlds.
2) *Unreal* - It is better suited to advanced users and has a higher learning curve because of its Visual Scripting and sophisticated graphical environment. It also provides BluePrints authoring tool which allows you to define the whole logic of the game using graphic diagrams, which can fully replace the requirement to write code in C++. Therefore, Unreal Engine is an excellent option for those that do not have programming expertise. This engine necessitates high-performance hardware, yet its visuals are outstanding.
3) *CryEngine* - CryEngine is a platform that should not be advised to anyone who has never created a game. A tiny number of tutorials and sloppy documentation are unlikely to entice new users. In comparison to other engines, it was primarily focused on the development of FPS games, maybe games of related genres. The game created in the CryEngine engine with the highest graphical settings performed the best in performance testing.
4) *GameMaker* - The core strength of GameMaker is in the creation of 2D games. It is really simple to learn. It takes practically no programming skills, which means that users without technical skills, such as designers or artists, may construct their projects without the assistance of a programmer using a drag-and-drop visual scripting tool, and its performance is at the highest level. One of GameMaker's greatest ironies is that, despite being designed to assist beginners to get things up and running quickly, there is no free version.

VI. Conclusion

In conclusion, this paper cannot suggest one game engine over another because each engine has pros and cons. The paper contributes to the field's literature by offering a comprehensive analysis of game engines that cater to various user profiles and demands. The option is determined by the user's profile and knowledge, the desired outcome, as well as their resources and time. The paper contributes to the field's literature by offering a comprehensive analysis of game engines that cater to various user-profiles and demands. Clearly, in terms of platform deployment criteria, Unity and Unreal are the best. However, when it comes to visuals and animation, Unreal

and CryEngine come out on top. A developer with a diverse set of talents should consider GameMaker and Unreal Engine. Making a decent game is not jeopardized by using the incorrect engine. They are only tools, the skill set of developers decides the game quality.

The comparative study of game engines discussed in our paper is limited to just four game engines that are widely used today. The paper lacks real-time implementation of games deployed via these game engines. The given parameters may cause conflict with the individual interests a game developer may have.

ACKNOWLEDGMENT

We would like to express our gratitude to our professor, Mrs Vijal Jain, who guided us throughout this research. We also wish to thank all students, game users and developers across the world for their research and contributions in the game development field, actively providing a deep insight in our comparative research.

REFERENCES

[1] Gregory, Jason. *Game engine architecture*. crc Press, 2018.

[2] Mishra, Prerna, and Urmila Shrawankar. "Comparison between Famous Game Engines and Eminent Games." *International Journal of Interactive Multimedia & Artificial Intelligence* 4.1 (2016).

[3] Maggiorini, Dario, et al. "Smash: A distributed game engine architecture." *2016 IEEE Symposium on Computers and Communication (ISCC)*. IEEE, 2016.

[4] Zarrad, Anis. "Game engine solutions." *Simulation and Gaming* (2018): 75-87.

[5] Pa.Megha L.Nachammai ,T.M.Senthil Ganesan "3D-GAME-DEVELOPMENT-USING-UNITY-GAME-EN-GINE" https://www.ijser.org/researchpaper/3D-GAME-DEVELOPMENT-USING-UNITY-GAME-EN-GINE.pdf

[6] Ahamed, Sagor et al. "Study Of An Application Development Environment Based On Unity Game Engine". International Journal Of Computer Science And Information Technology, vol 12, no. 1, 2020, pp. 43-62. Academy And Industry Research Collaboration Center (AIRCC), doi:10.5121/ijcsit.2020.12103. Andrade, Antonio. "Game engines: a survey." *EAI Endorsed Trans. Serious Games* 2.6 (2015): e8

[7] Nop, Kongdee, Jintapitak Manissaward, and Tangto Oattarapon. "Development of Character Design Frameworks using Game Engine: Unreal Engine." *2019 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT-NCON)*. IEEE.

[8] Torres-Ferreyros, Carlos Mauriccio, Matthew Alexander Festini-Wendorff, and Pedro Nelson Shiguihara-Juárez. "Developing a videogame using unreal engine based on a four stages methodology." 2016 IEEE ANDESCON. IEEE, 2016.

[9] Christopoulou, Eleftheria, and Stelios Xinogalos. "Overview and comparative analysis of game engines for desktop and mobile devices." (2017).

[10] Johnson, Claire. "Learning Basic Programming Concepts with Game Maker." *International Journal of Computer Science Education in Schools* 1.2 (2017): n2.

[11] "GameMaker," https://www.yoyogames.com/en

[12] Barczak, Andrzej, and Hubert Woźniak. "Comparative Study on Game Engines." *Studia Informatica. System and information technology* 1.23 (2019): 5-24.

[13] Computer Hope. "Device Driver" https://www.computerhope.com/jargon/d/driver.htm

[14] Satyabrata Jena. " Device Driver and It's Purpose". https://www.geeksforgeeks.org/device-driver-and-its-purpose/

[15] Deckhead. "Game Engine Devleopment:Engine parts" https://indiegamedev.net/2020/01/15/game-engine-development-for-the-hobby-developer-part-2-engine-parts/?openLinerExtension=true

[16] Unity "Unity User Manual" https://docs.unity3d.com/Manual/index.html

[17] "CryEngine," https://www.cryengine.com/

[18] "Unreal Engine," https://www.unrealengine.com/en-US/

[19] S. Brenden, Blueprints Visual Scripting for Unreal Engine: Build professional 3D games with Unreal Engine 4's Visual Scripting system, ed 1, (Jul. 2015).

[20] PV. Satheesh, Unreal Engine 4 Game Development Essentials, ed 1, United Kingdom (Feb. 2016).

[21] M. Menard, and B. Wagstaff, Game development with Unity. ed 2, USA (2015).

[22] A. Pattrasitidecha, Comparison and evaluation of 3D mobile game engines, Sweden (2014).

[23] Cruz-Neira, M. Fernández, and C. Portalés, Virtual reality and games, USA (2018)

[24] Carbonell-Carrera, Carlos, et al. "Think spatially with game engine." ISPRS International Journal of Geo-Information 9.3 (2020): 159.

[25] Pavkov, Sanja, Ivona Franković, and Nataša Hoić-Božić. "Comparison of game engines for serious games." 2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). IEEE, 2017.

[26] Armstrong, Mr Stuart. "Game Engine Review." NORTH ATLANTIC TREATY ORGANIZATION (2013).