COMPARATIVE STUDY OF GAME ENGINES

ESTUDO COMPARATIVO ENTRE *ENGINES* DE JOGOS

Carlos Gabriel Agostinho Gonçalo Klinke[*]
Adinovam Henriques de Macedo Pimenta[**]

ABSTRACT

The evolution of the game engines is now moving to more realistic and technically rich games in various fields such as physics, sounds and rendering. From the first engines to the most current engines for 3D gaming, the goal of development remains the same: to provide the developer with a platform to create exclusive games in such a way that developers do not have to write or develop the game from scratch, but only implement the idea with the help of an engine. In this way, many developers (experienced or beginners) have found in these tools the simplest, fastest and most professional way of developing new games. However, with so many engines available, the developer needs to make a choice considering the data most relevant to his project. This work analyzed data of several attributes relevant to the process of choosing considering the 10 most popular engines currently, serving as knowledge and aid to this decision making process.

Keywords: Engines. Games. Mobile. Console. Comparison. Creation.

RESUMO

A evolução das *engines* está agora se movendo para jogos mais realistas e tecnicamente ricos em vários campos, como a física, sons e renderização. Das primeiras *engines* até as *engines* mais atuais voltadas para jogos 3D, o objetivo do desenvolvimento permanece o mesmo: fornecer ao desenvolvedor uma plataforma para criar jogos exclusivos de tal maneira que os desenvolvedores não precisem escrever ou desenvolver o jogo a partir do zero, mas apenas implementar a ideia com a ajuda de uma *engine*. Desta maneira, muitos desenvolvedores (experientes ou iniciantes) têm encontrado nestas ferramentas a maneira mais simples, rápida e profissional de desenvolver novos jogos. Porém, com tantas *engines* disponíveis, o desenvolvedor precisa fazer uma escolha considerando os dados mais relevantes ao seu projeto. Este trabalho analisou dados de vários atributos relevantes ao processo de escolha considerando as 10 *engines* mais populares atualmente, servindo de conhecimento e auxílio à este processo decisório.

Palavras-chave: Motores de jogos. Desenvolvimento de jogos. Jogos digitais.

[*] Graduation in Computer Science at the Faculdade de Tecnologia, Ciências e Educação (FATECE). gabrielklinke@gmail.com
[**] Lecturer in Computer Science at the Faculdade de Tecnologia, Ciências e Educação (FATECE). adinovam@icmc.usp.br

Introdução

In recent years, the gaming market has been growing at a rapid pace. Games sales in 2015 reached a margin of US $ 91.5 billion, an increase of 11.84% over 2014. This caused companies to reach US $ 107 billion in 2017[1]. However, the growth is not only in sales, but also in the miscellany of games played, ranging from school games to first-person shooters (CHAUDY; CONNOLLY, 2018).

One of the factors that drive the growth of this miscellany is the increase in the number of game developers. This growth, which came mainly to the use of Game Engines for their development (HARDY et al., 2015).

A Game Engine, or simply engine, is a platform used for the construction of games and has the characteristic of facilitating the development of tasks related to games, such as: interpretation and execution of the game, calculation related and allows designers to focus on the subtleties that make the game inimitable, such as textures, relationship between game objects, sound effects, gameplay, and so on. (BISHOP et al., 1998).

The engines are a set of reusable modules that can be manipulated to take a game towards realism (LEWIS; JACOBSON, 2002). The main technical components of a game belonging to the engines are: graphics, animation, audio, physics, user interface, artificial intelligence, some codes already ready for certain actions such as skipping, running or attacking, and graphic contents such as scenarios and characters. The instantiation of these components, associated with the history of the game, the definition of its characters, the background, the avatar of the real world and its behaviors, etc., are the components that create the real game. In this way, we can see the engines as a middleware, offering all the necessary resources not only for the creation of the games, but also for the execution of them.

The term game engines originated in the mid-1990s and was used to classify games of companies that sold the cores (engines) of their games to other companies (BATTAIOLA et al., 2002). This nucleus began to be increasingly used with the growth of 3D games, such as DOOM and Quake, which are part of the games of the genre FPS (First Person Shooter, in free translation), because with the few technology that was then more convenient to use a ready core (TRENHOLME; SMITH, 2008).

[1] https://www.cnbc.com/2018/07/18/video-game-industry-is-booming-with-continued-revenue.html

Engines are the main format for building digital games nowadays, possessing a wide range of content library to aid in the development of game projects (COWAN; KAPRALOS, 2014).

With the growth of the gaming market growing every day, new game development tools have been emerging, confusing or even disrupting learning and game creation. In order to improve the technology, creation software is constantly being updated and, in a certain way, makes it difficult to choose a suitable tool for a particular game development project (CHAUDY; CONNOLLY, 2018). In this way, this work proposes a comparative study between the main engines with the intention of guiding the choice of the most suitable engine for the development of games by beginning developers.

1 General Objectives

The general objective of this work is to present a comparative study between the main engines available on the market today, specifying their characteristics in order to indicate which is the most suitable for a game project.

1.1 Specific Objectives

- Identify the main available engines;
- Identify the main companies supported by the engines, as well as the platforms and the programming languages supported by the engines.

2 Metodologia

In order to collect the data necessary for the development of this work, qualitative research will be used as a basis, according to Gerhardt and Silveira (2009, p. 37):

> It does not concern itself with numerical representativity, but rather with the deepening of the understanding of a social group, of an organization, and aims to generate new knowledge, useful for the advancement of science, without practical application envisaged. It involves universal truths and interests.

From the data collection, we will distinguish the attributes of the engines, being: platform compatibility; companies that use or use them; available versions (paid or free);

developmental languages; and focus of development (2D, 3D, Virtual Reality and Augmented Reality).

3 Evolution of Engines

In early 1989, a science fiction engine called Ultima Underworld was developed (BISHOP et al., 1998). This engine had the same name as the game itself. Later, after the launch of the Space Rougue game, Origin System had consolidated the Ultima Underworld engine in the market and developed the algorithm for texture mapping, which could be applied to floors, ceilings, walls, etc. (Dinken and McDowell, Johnson 2002).

In 1993, ID Software developed the Doom engine, which is not a 3D engine, but has the ability to represent objects, characters and an entire level map by representing 2D sprites (HERWIG; PAAR, 2002). The rendering was very fast by the standards of the time, and needed a 386-based PC with standard VGA support to run. Although it was a 2D engine, the illusion created by the developer made it a 3D title.

NovaLogic, in 1992, owned the Voxel engine, which was the basic engine of all games in the Comanche series (FRITSCH et al., 2004).

Voxel had its own way of representing volumetric objects, such as three-dimensional bitmaps. Before that, all engines applied vector graphics, which was a bit slower in speed and presented less detail compared to 3D bitmap presentation. Blade Runner, Command & Conquer are the featured games developed by the Voxel engine (KOT et al., 2005).

In late 1993, another game called Duke Nukem 3D was released on the market, having been developed with the help of the Build engine. Like the Doom engine, it also created 3D effects on the 2D interface simply by varying the sectors with different heights to achieve the illusion that the game was 3D. By applying special tags to multiple locations within a specific industry, developers can, whenever a player moves to these specific points, switch to a different industry, giving the illusion of changing levels or environment in time execution (CLUNT; BITTENCOURT, 2005).

XnGine, 1995, was the first 3D engine developed in the DOS operating system. Later, this engine started using high-resolution graphics and would be compatible with 3dfx video cards (MARKS; WINDSOR; WÜNSCHE, 2007).

The Quake engine, developed by Software ID in 1996, was the first truly 3D engine (MÓL; JORGE; COUTO, 2008). It had a unique processing capability for rendering maps by eliminating certain areas of processing that the player could not see. This comes from the fact that this engine uses Z-buffering, which is a method to determine which parts of the maps are visible to the player and render only those sections.

The 1996 Renderware engine was the most popular engine for cross-platform gaming. It supports the platforms PlayStation 2, Wii, GameCube, Xbox, Xbox 360, PlayStation 3 and PSP (FRIESE; HERRLICH; WOLTER, 2008).

The ID Tech 2 engine (known as Quake II), from 1997, has native OpenGL support, colorful lighting effects and C language support.

The 1998 GoldSRC engine took PC gaming into a new era supporting OpenGL and Direct3D. Some examples of successful OpenGL based games are: Half-Life, Day of Defeat and Counter Strike.

One of the most popular engines is the 1998 Unreal engine, which gave rise to the famous game Unreal Tournament. It integrates its own scripting language called UnrealScript and map editor called UnrealEd (PASSOS et al., 2009).

A modified version of the Quake II engine was the Quake III, designed in 1999, which supports 32-bit color, shaders, and advanced networking.

John Slagel, Red Faction's main programmer, developed the Geodmod engine (Geometry Modification) in 2001, with a focus on graphical reconstruction.

In 2001, the torque engine was developed to modify the FPS Tribes 2 game. It had an on-the-fly rendering option with fewer polygon counts and also had a world map editor integrated with it.

The 2001 Serious engine was designed to allow for large spaces and a large number of on-screen characters at any time and give rise to the popular Serious Sam series (HUDLICKA, 2009). Later, with Doom 3, 2004, most light surfaces were also made in real time, allowing for more realistic shadows, but at the cost of being able to render soft shadows. To get around this, projected lights could be used to create the illusion of soft shadows. 2005 Half Life 2 Source Creator Engine, including advanced shading technologies, dynamic shading and lighting, physics, and other various effects such as realistic-looking reflective water surfaces, real-time motion shadow, etc. (INDRAPRASTHA; SHINOZAKI, 2009).

The 2004 CryEngine engine used pixel shaders to get the realistic water effect on Far Cry. Developed by the company Crytek, she produced the game called Crysis, a DirectX 10 game.

4 Comparison Between Engines

The comparison between several game engines is not a simple task given the fact that there are several genres, types, multimedia support, middleware support, language and platform dependencies, rendering techniques and many other secondary resources (COWAN; KAPRALOS, 2014; SALEN; ZIMMERMAN, 2012).

One of the most recent works involving the comparison between engines considered six engines: Cry Engine, Hero Engine, Source Engine, Unity, Unreal Engine 3 and Vision Engine, which were the most popular in its time and used as a comparison criterion the creation platform , the supported programming language, and the physical library used (LEWIS; JACOBSON, 2002).

In this work, besides the characteristics used by Lewis and Jacobson (2002) will be considered: supported companies, graphic styles, support to augmented reality and virtual reality, and types of distribution, choosing the most used engines by companies and independent developers , according to Table 1.

Table 1. Engines chosen for the study

Engine
Unity
Construct 2
MonoGame
GameMaker Studio 2
Cocos2d-x
Godot Engine
Unreal 4
Amazon Lumberyard
Libgdx
CryEngine V

4.1 Business Support to the Engines

In addition to using engines for game development, many companies also use this tool to advertise their products in the commercial environment creating an interactive and fun advertisement for a particular product. An example of this is the games of the saga Lara Croft - Tomb Raider, whose first films leverage the game trade of the saga.

Following this idea, the prestige that the companies provide to the engines end up generating an incentive for the developers of these tools to improve their software.

Tables 3 and 4 list the companies supported by each engine. From these tables, a ranking with the engines of greater support to the companies was set up. This ranking can be seen in Table 2. The order in which the merged engines appear follows the order of greatest popularity (from highest popularity to lowest popularity).

We can see that Unity is the engine that has the most support for companies among the 10 most popular engines, followed by GameMaker Studio 2 and Cocos2d-x.

The larger the audience you want to achieve considering portability, the greater the number of companies supported by the engine. In this way, this ranking justifies part of Unity's popularity among game development teams.

Tabela 2. Ranking of the engines according to their support to the companies

Position	***Engine***	**Number of companies supported**
1ª	Unity	13
2ª	GameMaker Studio 2	11
3ª	Cocos2d-x	11
4ª	Godot Engine	8
5ª	Unreal 4	8
6ª	Amazon Lumberyard	8
7ª	Construct 2	7
8ª	MonoGame	6
9ª	CryEngine V	6
10ª	Libgdx	5

Table 3. Enterprise support given by Unity, Construct, MonoGame, GameMaker Studio 2 and Cocos2d-x engines

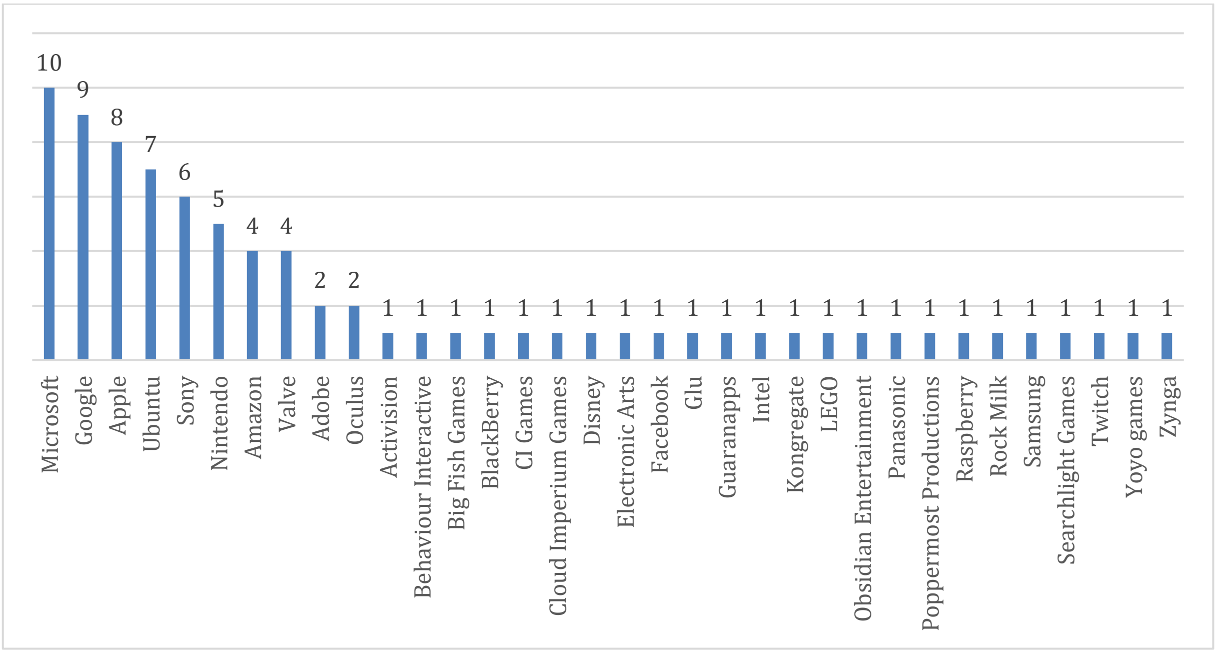
Unity	**Construct 2**	**MonoGame**	**GameMaker Studio 2**	**Cocos2d-x**
Google	Microsoft	Microsoft	Adobe	Google
Apple	Google	Ubuntu	Apple	Apple
Facebook	Amazon	Google	Ubuntu	Microsoft
Sony	Kongregate	Sony	Google	Ubuntu
Microsoft	Nintendo	Apple	Microsoft	Samsung
Nintendo	Ubuntu	Nintendo	Amazon	Panasonic
Adobe	Valve		Sony	Intel
Ubuntu			Nintendo	Zynga
Oculus			Raspberry	Glu
Electronic Arts			Valve	Big Fish Games
LEGO			Yoyo games	Disney
Ubisoft				
Square Enix				

Table 4. Enterprise support given by Godot Engine, Unreal 4, Amazon Lumberyard, Libgdx and CryEngine V

Godot Engine	**Unreal 4**	**Amazon Lumberyard**	**Libgdx**	**CryEngine V**
Microsoft	Microsoft	Microsoft	BlackBerry	Sony
Google	Apple	Sony	Google	Microsoft
Apple	Ubuntu	Twitch	Apple	Amazon
Valve	Nintendo	Apple	Microsoft	Poppermost Productions
Oculus	Valve	Google	Ubuntu	CI Games
Rock Milk	Sony	Amazon		Obsidian Entertainment
Guaranapps	Google	Cloud Imperium Games		
Searchlight Games	Activision	Behaviour Interactive		

Figure 1 shows the chart with the companies that have the most support for the engines. Microsoft ranks among the companies with the most support among the most popular engines, being supported by the 10 most popular engines.

Figure 1. Companies with more support by the engines



4.2 Creation Platforms

Creation platforms are the environments where the application will be developed. Some examples of platform include: smartphones, consoles, computers, etc.

As companies deployed their platforms for application, software and game development, a number of new companies emerged and new engines were created. With this, many opportunities were opened to small developers, beginners and enthusiasts, and were more likely to demonstrate their potential by dramatically increasing the demand for developers for the most diverse platforms.

Tables 5 and 6 show which creative platforms are supported by the respective engines.

Table 7 summarizes and ranks engines with the largest number of supported platforms. Once again Unity, this time tied with GameMaker Studio 2, stands out as the engine with the largest number of supported platforms.

Figure 2 shows the chart with the ranking of the platforms that have the support of the largest number of engines. Android and iOS stand out among the most supported platforms.

Table 5. Creation platforms supported by Unity, Construct, MonoGame, GameMaker Studio 2 and Cocos2d-x engines

Unity	**Construct 2**	**MonoGame**	**GameMaker Studio 2**	**Cocos2d-x**
Android	HTML5	TvOS	Adobe Flash	iOS
iOS	Windows SO	Android	Xbox One	Windows SO
Windows SO	Mac OS	Microsoft UWP	Windows SO	Android
PlayStation 4	Linux	Mac OS	Ubuntu	Mac OS
PlayStation 3	Nintendo Wii U	Linux	Android	Linux
BlackBerry 10	Xbox One	PlayStation 4	Mac OS	Tizen
Mac OS		Xbox One	HTML5	
Linux		PlayStation Vita	iOS	
Nintendo Switch		Nintendo Switch	Raspberry Pi	
Nintendo 3DS			PlayStation Portable	
Nintendo Wii U			Nintendo Switch	
Nintendo Wii			PlayStation 4	
Xbox 360			Microsoft UWP	

Table 6. Creation platforms supported by the Godot Engine, Unreal 4, Amazon Lumberyard, Libgdx and CryEngine V engines

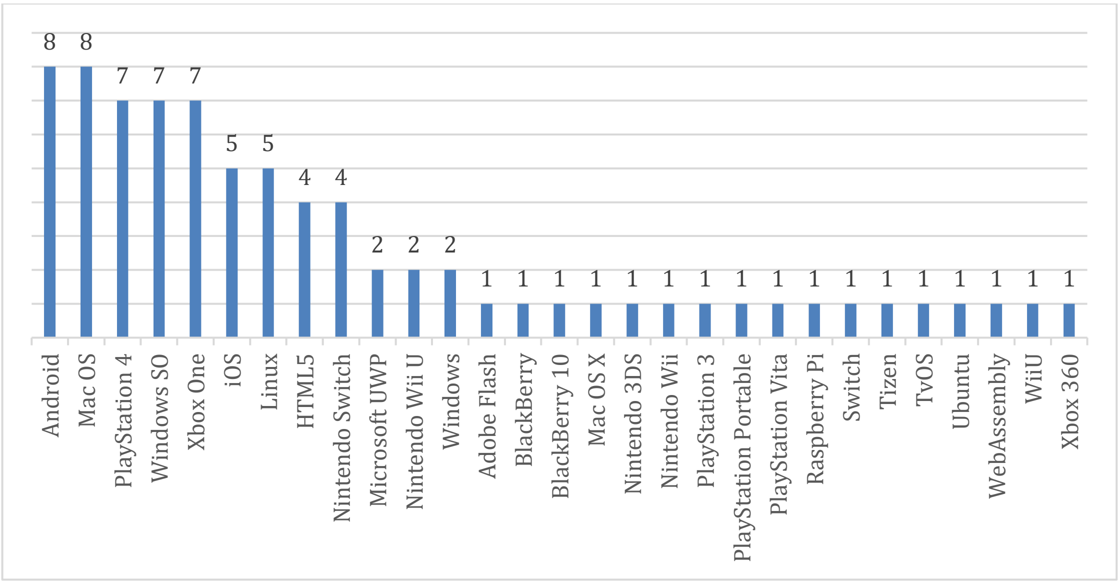
Godot Engine	**Unreal 4**	**Amazon Lumberyard**	**Libgdx**	**CryEngine V**
Windows SO	Android	Android	Windows	Windows
Android	Mac OS	Windows SO	Mac OS	Xbox One
Mac OS	Mac OS X	Xbox One	Linux	PlayStation 4
WebAssembly	Windows SO	PlayStation 4	Android	
PlayStation 4	Xbox One	iOS	iOS	
Switch	WiiU		BlackBerry	

Xbox One	PlayStation 4		HTML5	
	HTML5			
	Nintendo Switch			

Table 7. Ranking of engines according to their platform support

Position	***Engine***	**Number of supported platforms**
1ª	Unity	13
2ª	GameMaker Studio 2	13
3ª	MonoGame	9
4ª	Unreal 4	9
5ª	Godot Engine	7
6ª	Libgdx	7
7ª	Construct 2	6
8ª	Cocos2d-x	6
9ª	Amazon Lumberyard	5
10ª	CryEngine V	3

Figure 2. Platforms with more support by the engines



4.3 Programming Languages

For the complete development of the game, it is necessary to program the action of each object inserted in the game. For this, the engines provide by default support for programming languages. To these standard programming languages of the engines we call the primary programming language.

Through the creation of games and applications, and due to the language most used by students of development, a scope was created for the most diverse requirements assigned in a programmer. In this way, some engines allow coding of the game in programming languages that are not its primary language, such as Unity, which supports Java language, but whose compilation is done in C #, which is its primary language.

Table 8 and 9 report the primary programming languages of each engine and Table 10 ranks engines with the largest number of supported primary programming languages. In this new ranking we can see that the MonoGame and Cocos2d-x engines have the largest amount of primary programming language supported.

Figure 3 shows the graph with the programming language most used by engines as primary language. The C ++ language is the language most supported by the most popular engines on the market.

Table 8. Programming languages supported by Unity, Construct, MonoGame, GameMaker Studio 2 and Cocos2d-x engines

Unity	**Construct 2**	**MonoGame**	**GameMaker Studio 2**	**Cocos2d-x**
C#	JavaScript	C#	GML	C++
C++	C++	.Net	Java	HTML
		Java	C++	Lua
		C++		JavaScript

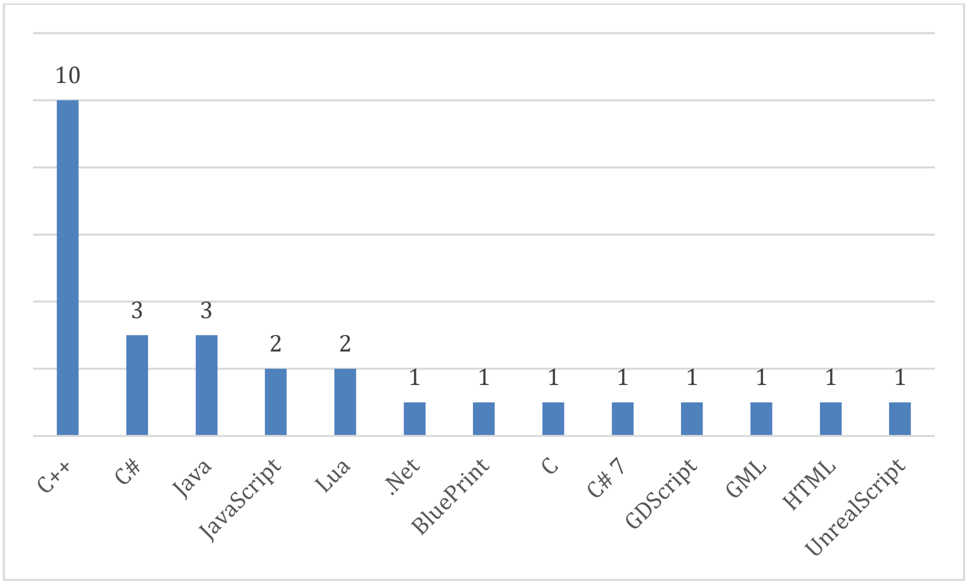
Table 9. Programming languages supported by engines Godot Engine, Unreal 4, Amazon Lumberyard, Libgdx and CryEngine V

Godot Engine	**Unreal 4**	**Amazon Lumberyard**	**Libgdx**	**CryEngine V**
C# 7	C++	C++	Java	C++
C++	BluePrint		C++	C#
GDScript	UnrealScript		C	Lua

Table 10. Ranking of engines according to the number of languages

Position	***Engine***	**Number of languages supported**
1ª	MonoGame	4
2ª	Cocos2d-x	4
3ª	GameMaker Studio 2	3
4ª	Godot Engine	3
5ª	Unreal 4	3
6ª	Libgdx	3
7ª	CryEngine V	3
8ª	Unity	2
9ª	Construct 2	2
10ª	Amazon Lumberyard	1

Figure 3. Languages with more support by the engines



4.4 Graphic Styles and their Demands

The demand for graphic game styles varies from project to project. In this way, several engines use the idea of being able to be implemented in any type of application, not limiting to only one graphic style. However, there are also unique environments for which they were developed. Thus, Table 11 shows which engines support 2D, 3D, or both.

Table 11. Engines for 2D and 3D environments

Engines 2D	**Engines 3D**	**Engines 2D/3D**
Construct 2	Amazon Lumberyard	Unity
Cocos2d-x	CryEngine V	Unreal 4
		GameMaker Studio 2
		Godot Engine
		MonoGame
		Libgdx

4.4 Virtual Reality and Augmented Reality

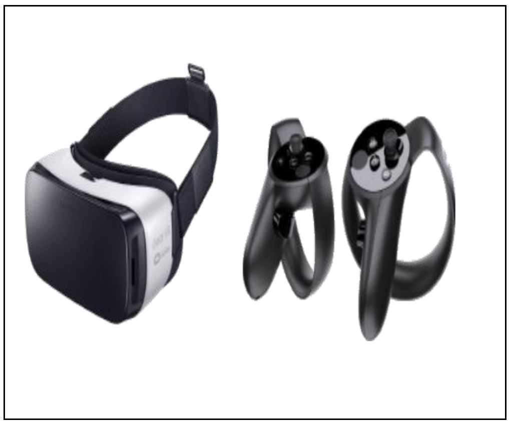
Virtual Reality (VR) is seen as an advanced interface used to access computer-based applications for visualization, movement and user interaction in real-time in three-dimensional computer-generated environments (KIRNER; SISCOUTTO, 2007).

VR is one of the technologies that is expanding nowadays. It provides the player's immersion into a new world using specific tools such as a VR goggles and controls (Figure 4).

As defined by Pimentel and Teixeira (1993):

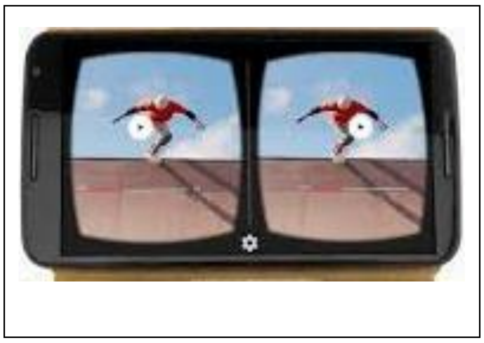
"*it is the use of high technology to convince the user that he is in another reality*".

Figure 4. Spectacles and control for VR



VR goggles work with mirrored images on two viewers positioned close to the eyes (Figure 5) causing the brain to interpret as a single image and also has a gyroscope in which the axis of rotation always remains in the same position making it possible to move the player's head to target the virtual world. The controls serve as auxiliary to the glasses, and may also have gyroscopes.

Figure 5. RV eyeglass mirrors

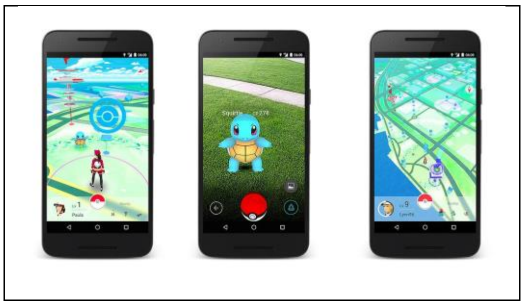


Augmented Reality (AR) is defined as the overlapping of virtual objects in the real world, through a technological device, improving or increasing the user's vision (AZUMA, 1997; BAJURA, 1995; KIRNER, 2004).

In addition to allowing this overlapping of virtual objects in the real world, RA also allows the manipulation of these objects with their own hands, enabling the user to have an attractive and motivating interaction with the environment (BILLINGHURST, 2001; KIRNER, 2004; SANTIN, 2008).

The RA works unlike virtual reality, because the game is performed in the real world through a screen, such as a smartphone, for example. An example of this game model is the Pokémon GO, which uses the camera of the smartphone to synchronize and project its creatures in our world in certain points of the region (Figure 6).

Figure 6. Illustration of GO Pokémon



Both VR and AR have been gaining popularity in the gaming market. Accompanying this popularity, the current engines already support the use of these technologies in their game development environment.

Table 12 shows how the carriers support the VR and AR support.

Table 12. Engines with VR and AR support

Engines VR	**Engines AR**
Unity	Unity
Unreal 4	Unreal 4
Amazon Lumberyard	Amazon Lumberyard
CryEngine V	

4.4 Development Versions

The versions of the engines influence the development of the projects that will be elaborated and several of the companies responsible create more of a version of the engines and some of them may be very different from each other. As an example of versions, we have some that can be paid and that include extra functions and tools that facilitate in the development. Because of this, it is up to the developer to decide which tool should work to facilitate and tailor their needs. Tables 13 and 14 show the free versions of each engines, while Table 15 shows the paid versions of some engines that charge for extra features.

Table 13. Free versions of Unity, Construct, MonoGame, GameMaker Studio 2 and Cocos2d-x engines

	Unity	**Construct 2**	**MonoGame**	**GameMaker Studio 2**	**Cocos2d-x**
Personal	✓	✓			
Educational		✓		✓	
Minimum					✓
Open Source			✓		✓
Free			✓		✓
Demo				✓	
Full					✓

Table 14. Free versions of engines Godot Engine, Unreal 4, Amazon Lumberyard, Libgdx and CryEngine V

	Godot Engine	**Unreal 4**	**Amazon Lumberyard**	**Libgdx**	**CryEngine V**
Personal					
Educational					
Minimum					
Open Source	✓			✓	
Free	✓	✓	✓	✓	✓
Demo					
Full					

Table 15. Engines with paid versions

	Unity	**Construct 2**	**GameMaker Studio 2**
Design - $			✓
Developer - $			✓
Console - $			✓
Professional - $	✓		
Plus - $	✓	✓	
Enterprise - $		✓	

5 Discussions

As seen in the previous section, the analysis of the 10 most popular engines involves features relevant to the process of choosing a particular engine. In order to make an overall analysis, we used some data from this research to assemble the Radar graph of these engines according to Figure 7.

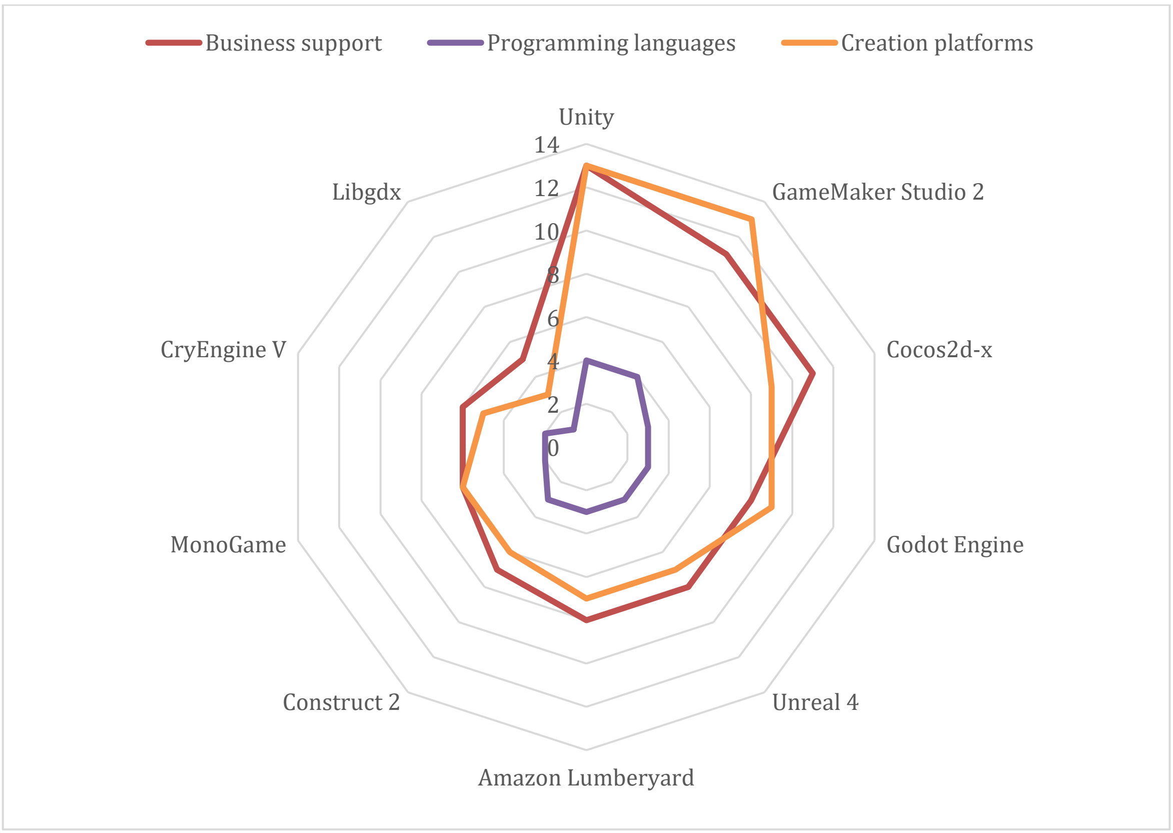
In this graph we considered the support to companies, platforms and languages, which are the characteristics used to assemble the 3 rankings of Section 4, shown in Tables 2, 7 and 10 respectively.

The graph of Figure 7 shows us a prominent slope for the Unity engine. Although GameMaker Studio 2 supports the same number of creative platforms, Unity wins in number of supported companies. This puts it in terms of reach that the projects made in this engine can achieve. As a negative point, the number of primary programming languages supported by Unity is among the smallest, decreasing the freedom of choice among programmers.

Unity can serve as a suitable tool for anyone who is starting to work with game development. In addition to its scope, this engine offers a free version for personal purposes, providing professional resources in an accessible way.

In the same way, the engine GameMaker Studio 2 presented good ranking. Although it has tied in relation to the number of creation platforms and lost to the Unity in the criteria of support to the companies, it has a programming language more than primary Unity. This can be a decisive factor for those who already have previous knowledge in JAVA or GML.

Figure 7. Overview of engines



Conclusions and Future Work

This work proposes to make a comparative analysis among the 10 most popular engines of the market, being able to serve as an auxiliary tool in the process of choosing for an engine among the game developers.

After identifying which are the 10 most popular engines currently, the research concluded that the Unity has the most number of characteristics considered important in the process of choice.

Another engine, GameMaker Studio 2, has also presented itself as a good choice for those looking for a complete game development environment.

As future work, we intend to analyze the computational performance of these 10 engines in two approaches: 1) computational cost (memory and processing expenditure) at development time; and 2) the computational cost of the games generated by these engines.

References

AMAZON LUMBERYARD, **Amazon Lumberyard – Game Engine**. aws.amazon.com/pt/lumberyard/, 2018. Available in: https://docs.aws.amazon.com/lumberyard/index.html. Access in: October 26, 2018.

AZUMA, R. T. A survey of augmented reality. **Presence: Teleoperators and virtual environments**, v. 6, n. 4, p. 355-385, 1997.

BAJURA, M.; NEUMANN, U. Dynamic Registration Correction in Video-Based Augmented Reality Systems. **IEEE Computer Graphics & Applications**, v. 15, n. 5. p. 52-60, 1995.

BILLINGHURST, M.; DUENSER, A. Augmented reality in the classroom. **Computer,** v. 45, n. 7, p. 56-63, 2012.

BISHOP, L. et al. Designing a PC game engine. **IEEE Computer Graphics and Applications,** v. 18, n. 1, p. 46-53, 1998.

CHAUDY, Y.; CONNOLLY, T. Specification and evaluation of an assessment engine for educational games: empowering educators with an assessment editor and a learning analytics dashboard. **Entertainment Computing**, v. 27, p. 209-224, 2018.

CLUA, E. W. G.; BITTENCOURT, J. R. Desenvolvimento de jogos 3D: concepção, design e programação. In: JORNADAS DE ATUALIZAÇÃO EM INFORMÁTICA (JAI), XXIV.; CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, XXIV., **Anais...** 2005. p. 22-29.

COCOS2D-X, Cocos2d-x – **Game Engine**. cocos2d-x.org/, 2018. Available in: http://docs.cocos2d-x.org/creator/manual/en/getting-started/cocos2d-x-guide.html. Accessed on: September 30, 2018.

CONSTRUCT-2 Construct-2 – **Game Engine**. scirra.com, 2018. Available in: https://www.scirra.com/manual/1/construct-2. Access in: October 05, 2018.

COWAN, B.; KAPRALOS, B. A survey of frameworks and game engines for serious game development. In: ADVANCED LEARNING TECHNOLOGIES (ICALT), IEEE INTERNATIONAL CONFERENCE ON, 14., **IEEE...**, 2014. p. 662-664.

CRYENGINE V, Cryengine V – **Game Engine**. cryengine.com, Available in: https://docs.cryengine.com/display/CEMANUAL/CRYENGINE+V+Manual. Accessed on: October 16, 2018.

DARKEN, R.; MCDOWELL, P.; JOHNSON, E. Projects in VR: The Delta3D open source game engine. **IEEE Computer Graphics and Applications**, v. 25, n. 3, p. 10-12, 2005.

ESA. **Essential facts about the computer and video game industry**. Retrieved August, v. 16, p. 2006, 2002.

FRIESE, K-W.; HERRLICH, M.; WOLTER, F-E. Using game engines for visualization in scientific applications. In: **New Frontiers for Entertainment Computing**. Springer, Boston, MA, 2008. p. 11-22.

FRITSCH, D. et al. Visualisation using game engines. **Archiwum ISPRS**, v. 35, p. B5, 2004.

GAMEMAKER STUDIO 2, Gamemaker Studio 2 – **Game Engine**. yoyogames.com, September 29, 2018. Available in: https://docs2.yoyogames.com/source/_build/index.html Accessed on: September 29, 2018.

GERHARDT, T. E.; SILVEIRA, D. T. **Métodos de pesquisa**. Porto Alegre: [s.n.], 2009.

GODOT ENGINE, Godot Engine – **Game Engine**. godotengine.org, October 2, 2018. Available in: http://docs.godotengine.org/pt_BR/latest/getting_started/editor/ Access in: October 2, 2018.

HARDY, S. et al. Framework for personalized and adaptive game-based training programs in health sport. **Multimedia Tools and Applications**, v. 74, n. 14, p. 5289-5311, 2015.

HERWIG, A.; PAAR, P. Game engines: tools for landscape visualization and planning. **Trends in GIS and virtualization in environmental planning and design**, v. 161, p. 172, 2002.

HUDLICKA, E. Affective game engines: motivation and requirements. In: INTERNATIONAL CONFERENCE ON FOUNDATIONS OF DIGITAL GAMES, 4., **Proceeding…** ACM, 2009. p. 299-306.

INDRAPRASTHA, A.; SHINOZAKI, M. The investigation on using Unity3D game engine in urban design study. **Journal of ICT Research and Applications**, v. 3, n. 1, p. 1-18, 2009.

KINER, C. **Projeto Sistema Complexo Aprendente**: um ambiente de realidade aumentada para educação (SICARA). 2007. Available in: http://www.ckirner.com/claudio/?PROJETOS:SICARA. Accessed on October 2, 2018.

KIRNER, C.; SISCOUTTO, R. A. **Realidade Virtual e Aumentada.** 2007. Available in: http://www.ckirner.com/download/livros/Livro-RVA2007-1-28.pdf. Accessed on October 2, 2018.

KOT, B. et al. Information visualisation utilising 3D computer game engines case study: a source code comprehension tool. In: SIGCHI NEW ZEALAND CHAPTER'S INTERNATIONAL CONFERENCE ON COMPUTER-HUMAN INTERACTION: MAKING CHI NATURAL, 6., **Proceedings…** ACM, 2005. p. 53-60.

LEWIS, M.; JACOBSON, J. Game engines. **Communications of the ACM**, v. 45, n. 1, p. 27, 2002.

LIBGDX, Libgdx – **Game Engine**. libgdx.badlogicgames.com, October 29, 2018. Available in: https://libgdx.badlogicgames.com/documentation/help/Documentation.html. Accessed on: October 29, 2018.

MARKS, S.; WINDSOR, J.; WÜNSCHE, B. Evaluation of game engines for simulated surgical training. In: INTERNATIONAL CONFERENCE ON COMPUTER GRAPHICS AND INTERACTIVE TECHNIQUES IN AUSTRALIA AND SOUTHEAST ASIA, 5., **Proceedings...** ACM, 2007. p. 273-280.

MÓL, A. C. A.; JORGE, C. A. F.; COUTO, P. M. Using a game engine for VR simulations in evacuation planning. **IEEE Computer Graphics and Applications**, v. 28, n. 3, p. 6-12, 2008.

MONOGAME, **MonoGame – Game Engine**. monogame.net, October 2, 2018. Available. In: http://www.monogame.net/documentation/?page=main. Access in: October 2, 2018.

PASSOS, E. B. et al. Tutorial: Desenvolvimento de jogos com Unity 3D. In: BRAZILIAN SYMPOSIUM ON GAMES AND DIGITAL ENTERTAINMENT, VIII. 2009. p. 1-30.

PETRIDIS, P. et al. Game engines selection framework for high-fidelity serious applications. **International Journal of Interactive Worlds**, p. Article ID 418638, 2012.

PIMENTEL, K.; TEXEIRA, K. **Virtual Reality – through the new looking glass**. New York: McGraw Hill,1993.

SALEN, K.; ZIMMERMAN, E. **Regras do jogo:** fundamentos do design de jogos. São Paulo: Blucher, 2012. V. 1.

SANTIN, R. **SACRA - Sistema de Autoria em Ambiente Colaborativo com Realidade Aumentada**. 2008. Available in: https://www.unimep.br/phpg/bibdig/pdfs/2006/DOIKPNGYIQHP.pdf. Accessed on October 2, 2018.

TRENHOLME, D.; SMITH, S. P. Computer game engines for developing first-person virtual environments. **Virtual reality**, v. 12, n. 3, p. 181-187, 2008.

UNITY3D, Unity – **Game Engine**. Unity3d.com, 26 de setembro de 2018. Available in: https://docs.unity3d.com/Manual/index.html. Accessed on: September 26, 2018.

UNREAL 4, Ureal Engine 4 – **Game Engine**. unrealengine.com, October 11, 2018. Available in: https://docs.unrealengine.com/en-us/. Access in: October 11, 2018.