INTERNATIONAL JOURNAL OF RESEARCH AND DEVELOPMENT IN ENGINEERING SCIENCES
e-ISSN: 2582 - 4201 Volume: 2 | Issue: 4 | Year: 2020

Warrior Queen: A Computer 3D AI Game

N P PATNAIK M [1] , BOSUBABU SAMBANA [2]

[1,2] Assistant Professor,
Department of Computer Science and Engineering,
Viswanadha Institute of Technology and Management, Visakhapatnam
Jawaharlal Nehru Technological University Kakinada, Andhra Pradesh, India

CH CHANDRA SEKHAR [3], G U V S VAMSI KRISHNA [4], CH SRINU [5], G KRUPA SYAM [6]

[3,4,5,6] IV B.Tech Students,
Department of Computer Science and Engineering,
Viswanadha Institute of Technology and Management, Visakhapatnam
Jawaharlal Nehru Technological University Kakinada, Andhra Pradesh

Abstract: - *The aim of our Project is to Design a game using unity 3D-game engine which is mainly focus on Gender Equality. The player plays as a* Queen*, who is going run and kill the enemies in the way to save the innocent people who had been exiled by enemies. The game is designed for Android and windows environment. In this project, we have used multiple built-in functions that unity 3D game engine provides.*

As a final product, we have developed a game that is fun and enjoyable, which converts the insights and thoughts of girls and women into audacity.

Key words: *3D, Computer Games, Software Engineering, AI*

1. INTRODUCTION

The game Warrior Queen is based on the concept of inequality between genders and it is designed on the platform of unity 3D engine. The game has two kingdoms (Player kingdom, Enemy kingdom), which are fight each other. In this war, the king of player kingdom was died, and peoples are trapped by against enemy kingdom. The player acts as a queen who is the daughter of king.

The player will release the trapped peoples from enemy kingdom by using the controls of queen. The application is a 3D-Android game Warrior Queen on his big Adventure. The Queen is not only smart but also a clever runner and she kills the enemies [1].

Computer games are programs that enable a player to interact with a virtual game environment for entertainment and fun. There are many types of computer games available, ranging from traditional card games to more advanced video games such as role playing games and adventure games. In this chapter, we first discuss the different types of computer games.

2. RELATED WORK

The architecture of computer games is also described. Finally, the programming environment that is used to build the computer games is discussed.

Types of Computer Games

Although computer games mainly provide entertainment and fun, it also improves hand/eye coordination and problem-solving skills. Each game has its own strategy, action and fantasy that make each game unique and interesting. Generally, we can classify computer games into the following types: card games, board games, puzzles, maze, fighting, action, adventure, role playing, strategy, sports and simulation games. However, the classification is a fuzzy concept, as many games are hybrids that fall into more than one class.

For example, Doom can be classified either as a maze game or an action game, while Monopoly can be classified as a board game or strategy game. The different types of computer games are briefly described as follows:

Card Games: They are computerized versions of traditional card games, or games which are essentially like card games in that they are primarily card-based (such as solitaire). Examples of card games include Blackjack, Bridge, Cisino, Solitaire and
.

Board Games: They are adaptations of classic board games. Examples of board games include Chess, Checkers, Backgammon, Scrabble and Monopoly.

Puzzles: Puzzle games aim at figuring out of a solution, which often involves solving enigmas, navigation, learning how to use different tools, and the manipulating or reconfiguring of objects. Mastermind and Tetris are examples of puzzle games.

Maze: Maze games require the successful navigation of a maze. Mazes can be viewed in different ways. For example, they may appear in an overhead view (as in Pac-Man), or first-person perspective (as in Doom).

Fighting: Fighting games involve characters who fight usually hand-to-hand, in one-to-one combat situations. The fighters are usually represented as humans or animated characters. Fighting games include Street Fighter, Avengers and Body Slam.

Action: Action games involve the human player shoots at a series of opponents or objects. Traditional action games include Space Invaders, Asteroids, etc. The recent popular action games are Doom, Quake, Descent, and Half-Life and unreal that involves the human player to control a character in a virtual environment to save the world from the forces of evil by using deadly force.

Simulation: There are two types of simulation games: management simulation and training simulation. Management simulation games refer to those games in which players must manage the use of limited resources to build or expand some kind of community, institution or empire. Example management simulation games include Railroad Tycoon; SimAnt, and SimCity.

For training simulation games, it refers to games that attempt to simulate a realistic situation, for the purpose of training. Through the game simulation, it helps the player to develop some physical skills, such as steering as in driving and flight simulation games. Example training simulation games include Police Trainer, Gunship and Flight Unlimited.

Game Design

A computer game can be just a C application program. The architecture of a typical computer game. It consists of the following components: Input, Game Logic, Graphics/Sound Support, Game Output and Networking. They are briefly described as follows: · Input – Users interact with the game program through input devices. Common input devices include keyboard, mouse or joystick.

Game Logic – It implements the game logic or game code that handles most of the basic mechanics of game. Generally, before the game logic is developed, the story line on how the game is played and how the players should interact should be designed. Simple physics, networking support and animations should be planned. In some advanced games, artificial intelligence (AI) and collision detection are also implemented in this step [2]. ·

Graphics Rendering Engine - It has complicated code to efficiently identify and render the game objects and background from a two-dimensional (3-D) model of the environment. It supports transformation of objects that are moved, rotated and scaled when required. ·

Graphics/Sound Drivers – The graphics drivers receive requests from the rendering engine to the graphics library using APIs. Windows APIs and Microsoft Foundation Classes (MFC) provide two-dimensional (2-D) graphics support for PCs. For supporting both 2-D and 3-D graphics, OpenGL and DirectX are the two most popular graphics libraries. DirectX also provides libraries for music and sound support. ·

Game Output – The generated 2-D or 3-D graphics is output to the display. The generated sound effect or music is output to the sound card. · Networking - It provides networking protocol support that allows

several users in remote locations to play and interact in the same game environment. In a networked game environment, a server is needed to maintain information on which the virtual game world is supporting, communicates with game clients that are used by players to provide them with information about the shared environment. The server also needs to synchronize the information, and maintain the consistent scenes of the virtual game world among the networked clients.

Programming Environment

Here, we intend to use computer games as programming examples to illustrate the different concepts in C such as branching, looping, functions, arrays, strings, and structures and file I/O. Advanced computer games such as role playing games, adventure games and simulation games require complex 3-D graphics to make the virtual game world realistic [3].

As such, only traditional, simple games that only require simple 2-D graphics such as drawing lines, rectangles and polygons are discussed. Windows API, MFC library and DirectX are only available in Windows platform, while OpenGL is an open source that can be available in both Windows and UNIX platforms. Here, we have chosen OpenGL and GLUT as the graphics driver for supporting different 2-D and 3-D graphics API for the developed game programs.

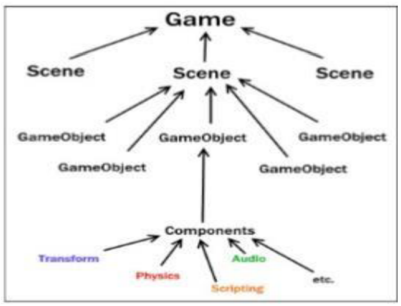
Microsoft's Visual C/C++ is used for the development of the game programs in the Windows environment. OpenGL and GLUT are required to be installed within the Microsoft's Visual C/C++ environment. We will also discuss the installation of OpenGL and GLUT for Microsoft's Visual C/C++.

How Unity Works?

In Unity, all game play takes place in scenes. Scenes are levels in which all aspects of your game such as game levels, the title screen, menus and cut scenes take place.

By default, a new Scene in Unity will have a Camera object in the scene called Main Camera. It is possible to add multiple cameras to the scene, but we will only deal with the main camera for now. The main camera renders everything that it sees or "captures" in a specific region called the viewport. Everything that comes into this region becomes visible for the player.

You can see this viewport as a grey rectangle by placing your mouse inside the scene view and scrolling down to zoom out the scene view. (You can also do so by holding Alt and dragging Right-click).



A scene itself is made out of objects, called GameObjects. GameObjects can be anything from the player's model to the GUI on the screen, from buttons and enemies to invisible "managers" like sources of sound.

GameObjects have a set of components attached to them, which describe how they behave in the scene, as well as how they react to others in the scene.

In fact, we can explore that right now. Click on the Main Camera in the Scene Hierarchy and look at the Inspector. It will not be empty now; instead, it will have a series of "modules" in it.

The most important component for any GameObject is its Transform component. Any object that exists in a scene will have a transform, which defines its position, rotation and scale with respect to the game world, or its parent if any.

The additional components can be attached to an object by clicking on Add Component and selecting the desired component. In our subsequent lessons, we will also be attaching Scripts to GameObjects so that we can give them programmed behavior [3].

Let us now consider a few examples of components – Renderer − Responsible for rendering and making objects visible. Collider − Define the physical collision boundaries for objects. Rigid body − Gives an object real-time physics properties such as weight and gravity [4].

- Audio Source − Gives object properties to play and store sound.
- Audio Listener − The component that actually "hears" audio and outputs it to the player's speakers. By default, one exists in the main camera.
- Animator − Gives an object access to the animation system.
- Light − Makes the object behave as a light source, with a variety of different effects.

In this chart, we can see how Unity composes itself through Game Objects into scenes.

The purpose of a literature review is to

- Provide foundation of knowledge on topic.
- Identify areas of prior scholarship to prevent duplication and give credit to other researchers.
- Identify inconstancies i.e. gaps in research, conflicts in previous studies, open questions left from other research.
- Identify need for additional research.
- Identify the relationship of works in context of its contribution to the topic and to other works.
- Place your own research within the context of existing literature making a case for why further study is needed.

The Unity game engine by Unity Technologies is the world's leading third party game making solution. It's the engine of choice for over 45% of the world's developers according to the Unity website's PR page.

This makes it by far the most used engine in the world. The same survey informs that their closest competitor (being Unreal Engine 4, or 'UE4', by Epic Games) has just over a third the number of users – 17% market share [5].

Unity engine is free and easy to install, and so anyone can have a fully featured game development engine installed on their Windows PC or Mac (Linux support is also on the way) with a minimum of fuss.

If you've never looked at an engine or line of code before, Unity's online tutorials can take you from your first moments in the editor and have you playing and sharing your very first game in just a couple of hours.

From there, you can complete more and more advanced tutorials until you're playing multiplayer co-op games against AI enemies [7].

Cost

- Unity Personal is both free and fully featured, allowing developers and students to be creative and drive the gaming industry forward.

Coding

- While Unity's framework is built on the C++ language, users interact with the engine through either C# or JavaScript.
- While these are fairly easy to learn and well supported by users on the Unity Answers forums, using Unity won't expose you to C++ code which is a big consideration if you want to specialise as a games programmer in the industry at large.

Memory Management

- One of the major benefits of using Unity is that a lot of the advanced engine coding is handled for you.

- If you're programming in C++ on another engine you'll need to specify what to do with garbage collection and memory allocation, but not in Unity.

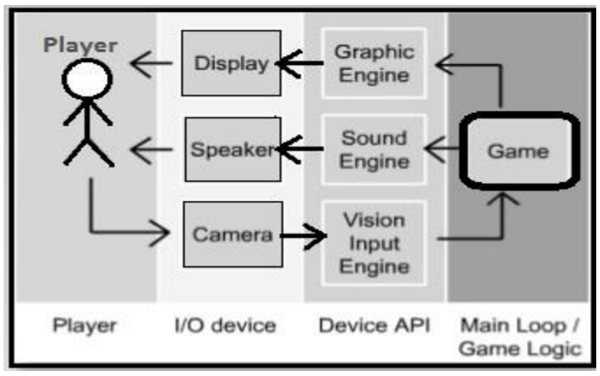
- You've less to worry about, and so you can prototype faster and generate fewer bugs, but you also have less control.
- This is a key area to consider when choosing an engine for your game.

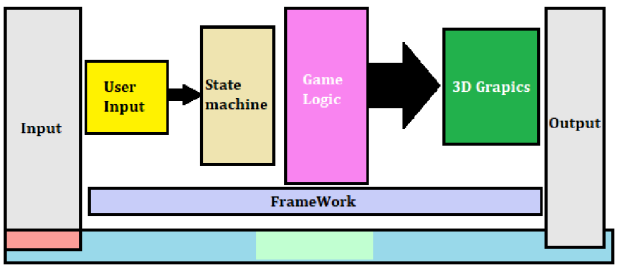
- Most terms will happily take the added speed and convenience, but this can be a deal breaker for those that want absolute authority over the engine.

Graphics

- I mentioned above how developers have tended towards UE4 and CryEngine if they wanted their game to look beautiful, but since Unity added PBS and other lighting improvements.
- At a very simple level, Physically Based Shading/ Rendering (PBS/PBR, same thing) is designed to give more realistic lighting to environments by treating metallic and non metallic surfaces differently in the lighting calculations.

The architecture and structure of a game is similar to that of software. But it does have some additional components which makes it different from software. Every game has the following components:





- Graphics Engine
- Sound/Audio Engine
- Rendering & Vision-Input Engine
- I/O Devices (like, Mouse, keyboard, speaker, monitor etc)
- DLL files and Drivers/Device APIs

Unity 3D

Unity is a cross-platform game creation system developed by unity Technologies, including a game engine and integrated development environment (IDE). It is used to develop video games for web sites, desktop platforms, consoles, and mobile devices. Unity is notable for its ability to target games to multiple platforms. Within a project, developers have control over delivery to mobile devices, web browsers, desktops, and consoles. Supported platforms include BlackBerry 10, Windows Phone 8, Windows, Android, iOS.

Visual Studio

Visual Studio is an Integrated Development Environment (IDE) developed by Microsoft to develop GUI (Graphical User Interface), console, Web applications, web apps, mobile apps, cloud, and web services, etc. With the help of this IDE, you can create managed code as well as native code.

Unity – using the Asset Store

The Asset Store is one of Unity's greatest strengths in the game engine market; it comprises a large number of assets, tools, scripts and even entire readymade projects for you to download.

In this example, we will be importing the Survival Shooter Tutorial project. To do so, we will search for it in the tab, and click on the asset published by Unity

System Testing and Test Cases

Game testing, a subset of game development, is a software testing process for quality control of video games. The primary function of game testing is the discovery and documentation of software defects.

The Unity Test Framework package is a tool that tests your code in both Edit mode and Play mode, and also on target platforms such as Standalone, Android, or iOS.

Process : A typical bug report progression of testing process is seen below:

Identification: Incorrect program behavior is analyzed and identified as a bug.

Reporting: The bug is reported to the developers using a defect tracking system. The circumstances of the bug and steps to reproduce are included in the report. Developers may request additional documentation such as a real-time video of the bug's manifestation.

Analysis: The developer responsible for the bug, such as an artist, programmer or game designer checks the malfunction. This is outside the scope of game tester duties, although inconsistencies in the report may require more information or evidence from the tester.

Verification

After the developer fixes the issue, the tester verifies that the bug no longer occurs. Not all bugs are addressed by the developer, for example, some bugs may be claimed as features (expressed as "NAB" or "not a bug"), and may also be "waived" (given permission to be ignored) by producers, game designers, or even lead testers, according to company policy.

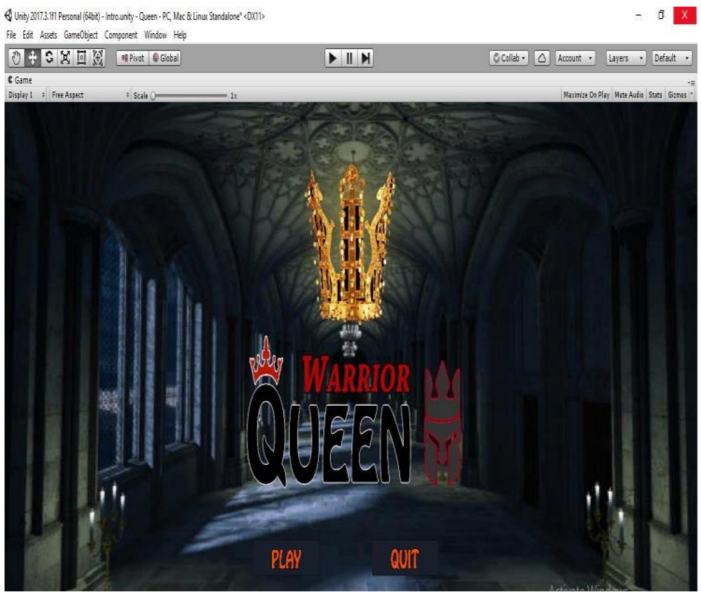
Methodology

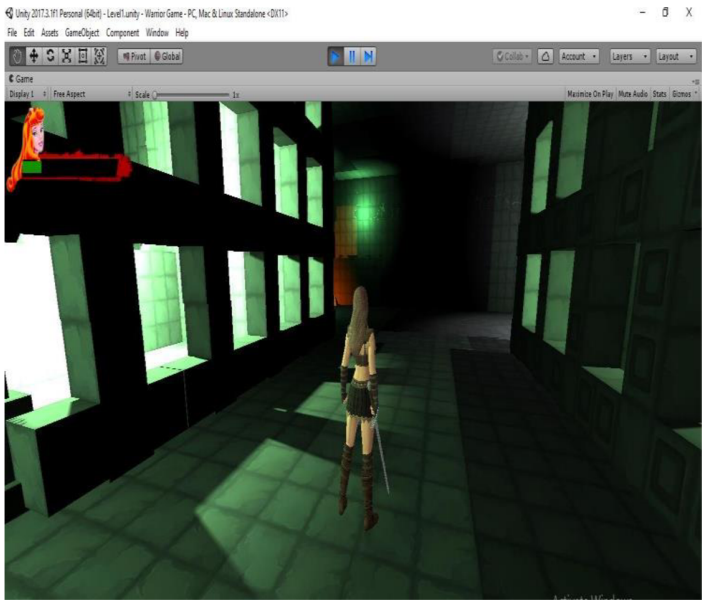
There is no standard method for game testing, and most methodologies are developed by individual video game developers and publishers. Methodologies are continuously refined and may differ for different types of games.

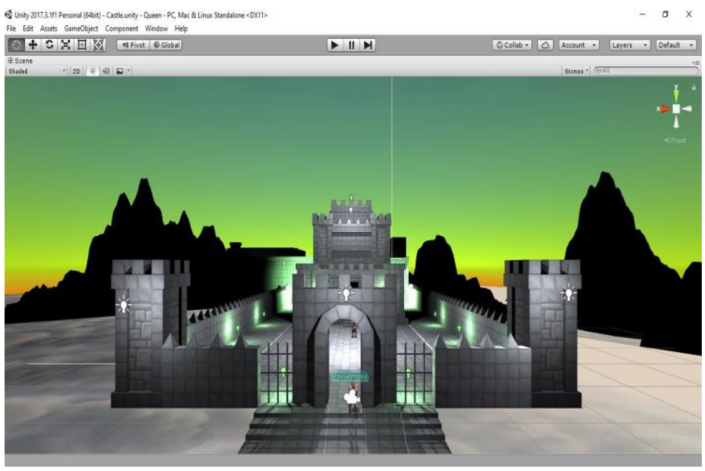
Many methods, such as unit testing, are borrowed directly from general software testing techniques. Outlined below are the most important methodologies, specific to video games.
Functionality testing

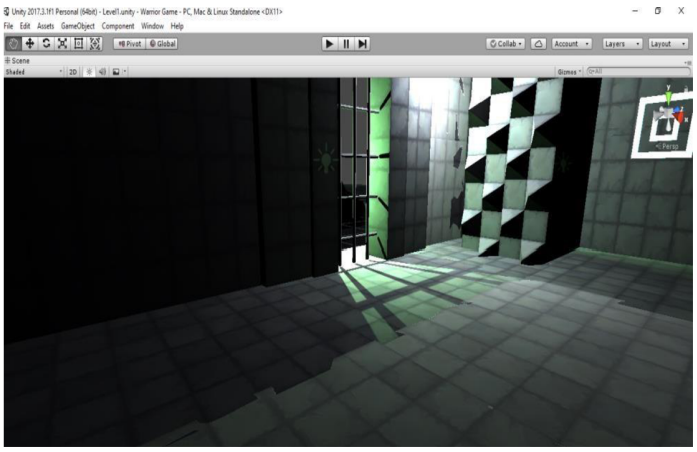
- Compliance testing
- Compatibility testing
- Localization testing
- Soak testing
- Beta testing
- Load testing
- Multiplayer testing.

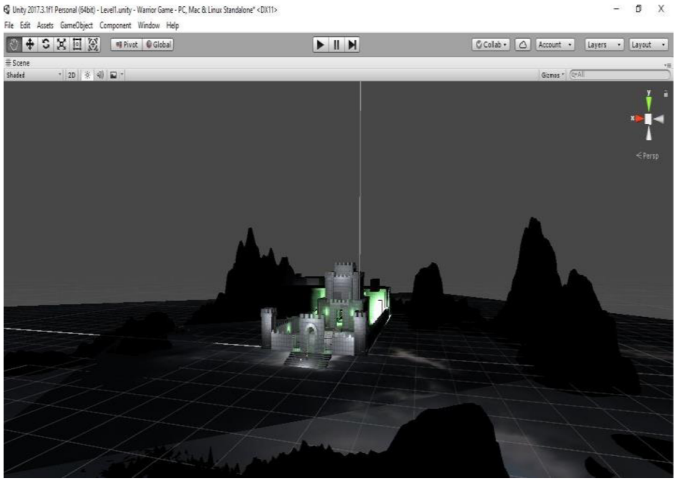
RESULTS

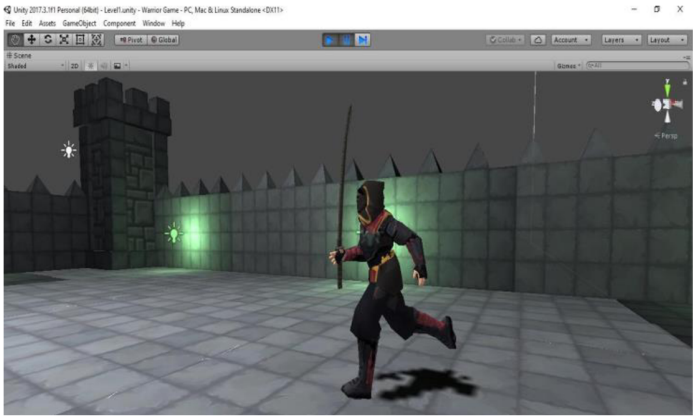












3. CONCLUSION

In this paper, we have suggested Data mining technology provides a whole new way of exploiting large databases. In the wrong hands, this can be a problem. However, it is not the tools themselves, but the combination of tools and data that pose a problem. The obvious solution is to restrict access to data. Many data mining techniques can be implemented data to predict their future performance.

A Software project means a lot of experience. In this section we summarize the experience gained by project team during development of "Warrior Queen".

Evaluation of objective and AIM

The main objective is

- To which are implemented in this project are following.
- To build an interesting game for user to play.

- To build a game that improves the decision-making power of a user as he/she has to make a decision quickly.

The Obstacles

- Working with Unity game engine completely a new experience for us. Normally we are working with different OO languages, DBMS, mark up languages etc.
- We adapt these things by video tutorials, text tutorials, Internet and learning materials given by the tools themselves. It's a matter of time, patience and hard work.
- It is very sensible work and it demands much time because the game engines try to connect game environment with the real world.
- Creating a 3D model is very difficult because you need to work with each and every point of the model.
- After all the things is that a game project is not a project of 6 months for four people!

The Obstacles

- Working with Unity game engine completely a new experience for us. Normally we are working with different OO languages, DBMS, mark up languages etc.
- We adapt these things by video tutorials, text tutorials, Internet and learning materials given by the tools themselves. It's a matter of time, patience and hard work.
- It is very sensible work and it demands much time because the game engines try to connect game environment with the real world.
- Creating a 3D model is very difficult because you need to work with each and every point of the model.
- After all the things is that a game project is not a project of 6 months for four people!

The Achievements

- Now we know much more about game engines. How it works? The properties, objects and others. We know how a model is constructed and how it is animated.
- The main thing is that as a software engineer, skill and expertise to create a SRS document and an overall software product report is now better than before.
- Co-Operation between group members.
- To Develop Communication Skills.
- Growing critical thinking and imagination capability.
- We learned a lot through this project. This project has sharpened our concept of game engine, animation and the software-hardware interface.

Future Enhancements

- Level Extension
- Improve Graphical representation
- Introduce new game features
- Introduce new environment and scenes
- Take user response through website.

REFERENCES

[1]. Bosubabu Sambana, "Blockchain Approach to Cyber Security Vulnerabilities Attacks and Potential Countermeasures", International Journal of Security and Its Applications (IJSIA), ISSN: 1738-9976(Print); 2207-9629(Online), NADIA, Volume 14, Number 1, 2020 March, pp. 1-14.

[2]. R. M. Ryan, et al., "The motivational pull of video games: A self-determination theory approach," Motivation and emotion, vol. 30, pp. 344-360, 2006.

[3]. K. A., "Ibisworld: Video games in the US," 2012.

[4]. D. Takahashi, "Time spent playing video games keeps going up," 2010.

[5]. K. Roe and D. Muijs, "Children and computer games: A profile of the heavy user," European Journal of communication, vol. 13, pp. 181-200, 1998.

[6]. M. Seifi, et al., "The Effect of Computer Games on Students' Critical Thinking Disposition and Educational Achievement," International Journal of Education & Literacy Studies, vol. 3, p. 36, 2015.

[7]. M. Salimi, "The Effect of Computer Games on Students' Performance in Mathematics," Mediterranean Journal of Social Sciences, vol. 7, p. 157, 2016.