SCHOOL OF ARCHITECTURE, COMPUTING & ENGINEERING

BSc in Computer Science

Michail Markou

CN6000 – Mental Wealth 3

UEL NUMBER

2020732

Date

11/11/2021

Advanced Gameplay Survival Mechanics First-third/person for multiplayer video games

Michail Markou

University of East London

Contents

1. Abstract

Video games are high-involvement products that tend to retain their players thought out the duration/narrative or objective goal[1]. At its core of implementation from business goals to product delivery, there is where the gameplay systems sit.

Because of a plethora of game genres categorization even though the line can be blurred easily between each genre this is dependent on the project's vision; this study will be focused on survival experience multiplayer horror approach with FPP (first-person perspective) and some RPG (role-playing) mechanics mixed.

Our thesis project study addresses this succession line of which games are being made in the literature by analyzing the core concept and actual implementation of an immersive virtual world of innovation, freedom and a form of digital identity of an offline or online player instance.

First, we analyze the observable factors, a dataset and impact of 3 well known multiplayer co-operative and non-co-operative video games via Steam Charts associated with a similar genre approach. The historical behavioral data and the time of the game's actual release throughout its competitors and deconstructing the mechanics/systems in short of each game and why it retains the majority of the players more than the others.

We also show that player retention diminishes due to the absence of a particular game mechanic or lack of a plethora level design strategy that contributes to the factors as well.

Keywords: Game Design; Multiplayer Games; Player Behavior; Player retention; Gameplay Mechanics; Gameplay Systems;

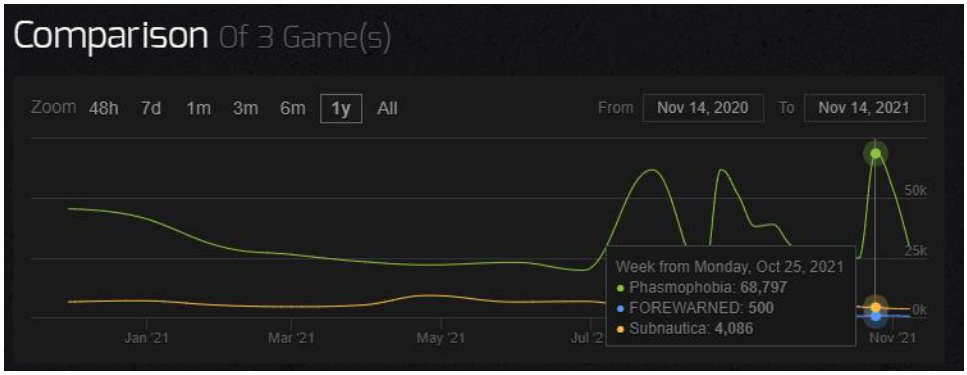

Figure 1 https://steamcharts.com/cmp/739630,1562420,264710#1y

[1] Either keeping at the edge of non-linear narrative perspective of Hero's outer and inner Journey aka from "real" world obstacles to overcome or fail; all the way through identity to the essence or by introducing non-story multiplayer skill competence and attributes or else combination of everything above.

2. Introduction

2.1. What are Video Games

A video game on the other hand is a game played by electronically manipulating images produced by a computer program on a monitor other display. [1]

Question: What does this mean?

Answer: A huge amount of effort to bring real-world data,

about accurate physically world (or not) to a digitalized form representation.

That immediately makes gameplay mechanics a very dependent thing. Agnostic Creation and abstraction of such a system are really difficult to make.

We must be also aware of other fields in form of assets such as AI, Sound, 2D/3D art, Animation, visual effects, rigs, shaders, Graphics, textures, physics and any other form of external Data like CSV/excel files or any proprietary extension that can be translated and be understand by a game engine.

2.2. Influence

The “video game” industry represents one of the most significant pillars/components of the global market expanded in many fields, e.g., entertainment, training and simulation, architectural and automotive visualization, higher education, linear film and television content creation, broadcast and live event production, real-time virtual production, metaverse and other real-time applications. According to Europe’s video game industry (ISFE), consumers reach a 50% mark of the European population aged between 6-64. The average age between them is 32 years old and half of that population, the 47%, are women’s across these markets. Usually, there is a higher chance of someone pursuing a STEM job when playing video games, which concludes in higher science development rates.

Also, another excellent impact point is that 10h/week the average is being spent playing video games. In contrast, 14/h week on social media and 24/h week on watching tv from these statistics can someone easily tell that this is all part of the “video game” industry in a way, the virtual world is everywhere and its growing onset rapidly [2].

Now you may wonder where gameplay mechanics fit here? In a broader term, well actually everywhere because they define the field and ruleset of the deterministic non-linear virtual or augmented reality world. Everything has its logic and with the logic comes the actual design and implementation of gameplay systems of interactions because it is nothing more than what a user will input as his next instruction set in the iconic or more pronounced virtual world, nowadays they can be expanded upon what’s called gamification, this concept takes the systems in a game and applies it in real-world data that way because of its nature, a game always pushes its player to achieve something so players get rewarded[2] [3]. The world is digital, and life events also contribute to these areas. As pandemic hits us or world pollution and climate change become a more controversial issue for our day’s digitalization is a must and that means at its core, “gameplay” mechanics usage, proper structure, architectural software solutions are becoming our reality, modern problems require modern solutions.

From an entertainment perspective, modern video games are high-involvement products with emergent multiplatform and multiplayer skill and attribute characteristics or story-wise, aiming to deliver long-term happiness to consumers [3]. This directly leads to more significant retention of consumers in the market, which is often seen as more preferable and profitable than acquiring new ones. Jolley et al. argue that retention can be measured by the duration of time a consumer continues to buy from a company [4]. Rust and Zahorik add that retention can be viewed as the propensity for a consumer to stay with a brand over time [5]. To improve player retention rates beyond the short scope term, producers attempt to efficiently organize and effectively create immersive blueprints for match participants in the multiplayer world into teams and thus customize the video game experience around aspects of the player such as preferences, playing style and skill level [6][5.1].

What makes a game appealing? Is it the story? Is it just for fun? Do we like spending more on thinking while playing or just playing for nothing?

Consequently, this study addresses three key research questions related to enjoyment, matching and retention of players in multiplayer video games:

[2] We see this almost always in the Internet many web sites provide unlockable achievements when you doing something based on their “business” rules or it could also be applied in higher education which is a type of a Serious Game.

[3] High scores create the charm of repetition [14]

2.3. Research questions:

(i) Which genre affects the most a player by design?
(ii) Which observable, game logic behaviors tend to affect player retention?
(iii) What is the proper way of building an architectural gameplay system?

We address these research questions through a multi-stage analysis approach.

First, we go through comparing an ongoing analysis of Steam's concurrent players using its dataset. Following the extensive empirical analysis, we dig into which game core ideas of systems can influence a game the most such as teamwork vs game difficulty ratio balance. Then based on findings, we conclude what is the best way to develop similar systems and structure architecture in order to create an immersive genre video game experience by reverse-engineering the existing ones.

3. Literature Review and Hypotheses.

3.1. Game Design idea principles

The late 70s were interesting for many reasons. That high (g)old retro era is still at the top despite the year goes. It has nothing to do with what you put on the screen or the game Design is what happens in the players' heads. Making a TV screen from a passive medium to an active medium by re-imagining the world is a success [7]. Space invaders among others launched back then. This is the first game that has actual AI and you have to have a strategy to achieve a high score. To beat its level, you must recognize patterns of artificial intelligence to "reverse-engineer" it [8, 9]. Games have suddenly become so immersive because of the "flow" which is known in many fields among gaming which is something so challenging fun and addictive that you focus on it and lose track of time [10]. In addition, the whole "script" of the game should remain unpredictable with a stochastic process technique but not all of its systems [11]. A Game must be biased in the player's favor.

There are 3 big principles for a game design (aka is my game fun):

1) Build around a core game mechanic

The best way to understand something is by studying something similar.

Grab a concept mechanic and make it to last the entire duration of game e.g., in Portal video game players have a Portal gun that they are using to solve puzzles.

If this mechanic, which players will be performing constantly during your game, is uninteresting, your design has failed. Even if you repeat something make it fun

by introducing new elements such as new abilities, new enemies, new harder platforming sections.

2) **Easy to learn but fun to master**

It must have a depth to it no matter if it's competitive or not but each section must behave with a pattern that by repetition could be recognized analyzed and solved by the player.

3) **Reward the player**

Depending on the content as human beings we like getting feedback from our hard work actions. Give players something like easter eggs, hidden levels, new abilities, secret boss fights or secret cutscenes [12].

3.2. Methodologies | Objectives

The best toolset for a 3D world and a Lead industry Standard in many fields but in our case for console/PC gaming is Unreal Engine.

3.2.1. Features of the system/s

Below we see a high-level content of what the system is apart (class style hierarchy).

To build a system, you must first understand the *style, taste, vision direction* of specific game genres it cannot have everything if it doesn't fit the style direction itself.

Version 1:

1. Player:
    i. Health System
    ii. Sprint/Stamina System
    iii. Damage System
    iv. Dynamic Inventory System as component
    v. Footstep Sound System (with physical materials)
    vi. Full Editable Inspection System as component
    vii. Realtime Depth of Field
    viii. Diary (Quest) System as component
    ix. Interactable Physics System
    x. Save System as component
    xi. User Interface/Widget System
    xii. NPC behavior
2. Flicker System as Component (attached to other Actors e.g., Light Actors)
3. Light Actors Equipment:
    i. Flashlight System
    ii. Flare System
    iii. Candle System

iv. Camcorder and polaroid System
    Night Vision System
    Record System
    Photo Capture System (interact with environment objects)
v. Torch System
vi. Glowstick System
vii. Lighter System

4. Save System
    i. Local save files
    ii. Database
5. Animation System
    i. Player Locomotion
    ii. Light Actors

3.2.2. Software Implementation Architecture

Unreal Engine provides many different approaches when comes to software engineering.

It gives you two options for writing code:

1) **Blueprint Programming**: Design Level abstraction (visual scripting/programming) such as UML mixed with actual implementation code on that level by connecting nodes[4] (running over virtual machine-like java does by converting it to intermediate instruction set then to machine code) (*slower compile-time Faster Development time, and a performance hit on the runtime of the game*[5]).

    1.1) **Nativization of Blueprints** (Reduces VM overhead): Hybrid approach by converting before cooking of the final game code from Blueprints to C++ with some overhead extra code auto-generated but the gain is 90%+ compared to Blueprint approach [15].

2) **"Coding":** text editor style like C++. Code Runs from C++ to machine code pipeline (*faster compile time Slower Development time. Faster execution at runtime of the game*).

3.2.3. Object-Oriented Programming

After of Software implementation architecture (in the previous step) that has been selected, we proceed with how we build the system.

[4] It retains all the concepts of Software Engineering architecture despite that has a node graph. In the end of the day abstraction level is what we are trying to achieve e.g., from circuit switching to assembly to C++ all the way to python with GitHub Copilot AI. It just makes you more productive and more valuable as an asset of the business.

[5] Nowadays Consoles/PC's can easily handle raw Blueprint code without any slowdown.

The system follows a modern approach, a hierarchical pattern of classes with inheritance. Everything is an object[6] and each Object actor can have inside him another object like a sub-object. Every gameplay mechanic will be implemented with this in mind[7].

Communication between objects/Blueprints is done via 2 ways.

1) Interfaces
2) Casting

When an object of a different hierarchy wants to communicate with another object then interfaces will be used. If it is in the same hierarchy casting of an object from lower to higher and the inverse in the hierarchy can be used as an alternative that's how you get parents properties or methods to be executed[8].

3.2.4. Server-Client Architecture

Making a Network replicated aka multiplayer game in Unreal Engine code is actually all code convention meaning that is not an option you check and activate but a coding style you follow, this also acts as security control/measure for instance a player/client can execute malignant/cheat code of the game but if this is not matching in servers side the client's code then it won't be executed and the player will create "laggy" results on the game thus it can be detected and kicked out/banned from the game.

You need to tell each game instance to send information to the server and the server will multicast back to all client player instances. Not all information is necessary to be known from the server because not all clients should know it either. If its client only like what happens on your screen menu it's per player locally but your health for example needs to be updated among other clients as long as your screen too.

Another Aspect of Server-Client architecture is Save Game Component. Traditionally you would create a local save game file on each computer and each of these computers will store all the data about the player's position in the world, the status of the player and inventory to load and other variables to spawn again but the drawback is if any computer loses a file, then that player will be forced to play from the beginning in order to unlock all the perks.

[6] e.g., Actor of Player, bullets, flashlights, doors, lights, windows, wall, vehicle etc.

[7] e.g., player moves, player shoots, player interacts with the environment and the environment (object/s) respond back from caller to callee and the inverse.

[8] In the above example we describe communication between 2 actor references/objects always from a third object/actor reference point of view.

The solution to this is to create a database like MySQL for persistent storage of information like a save game either in a private cloud e.g., a supercomputer or public one like AWS with Kubernetes instance for dynamic resources to allocate.

3.2.5. Game AI

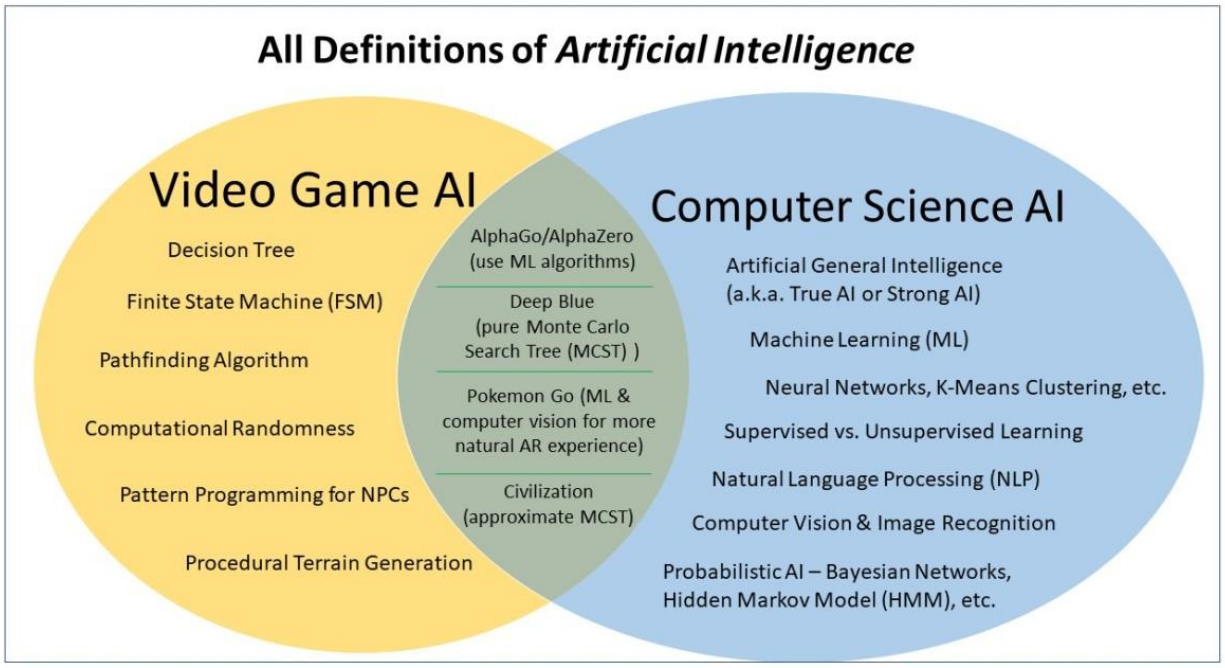

Figure 2 https://videogameintelligence.com/

Like everything else in-game design AI must fit the game's intended experience there is not a common AI to fit in all games it must do more than just kill the player it must be ambitious and ingenious. Actually, nowadays AI differs from academic AI. It serves to improve the game-player experience rather than machine learning or decision making. During the golden age of arcade video games, the idea of AI opponents was largely popularized in the form of graduated difficulty levels, distinct movement patterns, and in-game events dependent on the player's input. Modern games often implement existing techniques such as pathfinding and decision trees to guide the actions of NPCs. AI is often used in mechanisms that are not immediately visible to the user, such as data mining and procedural-content generation.

However, "game AI" does not, in general, as might be thought and sometimes is depicted to be the case, mean a realization of an artificial person corresponding to an NPC, in the manner of say, the Turing test [13].

3.2.6. Game Mode

The game mode is a way to structure our game logic without using too much repetitive code (e.g., Level blueprint) and transfer it to new levels and maps.

You can have a different game mode for single player and multiplayer for example that way we define clearly which code is going to be executed on its player choice.

3.3. Expected results

- The player can save the game
- Players can see each other interactions/animations without lag (Network replicated)
- The Player can use Equipment
- An NPC AI can hunt the player.

References

Academic writing:

[1] Google Search: What is a video game, Definitions from Oxford Languages.

[2] Europe's video game industry, 2021 key facts about the European video games sector, available at https://www.isfe.eu/data-key-facts/key-facts-about-europe-s-video-games-sector/ Accessed: 11th November 2021.

[3] E. W. Anderson, C. Fornell, D. R Lehmann, Customer satisfaction, market share, and profitability: Findings from Sweden, Journal of Marketing V.58 (N.3) (1994) pp. 53-66.

[4] B. Jolley, R. Mizerski, D. Olaru, How habit and satisfaction affect player retention for online gambling, Journal of Business Research 59 (6) (2006) 770 – 777.

[5] R. T. Rust, A. J. Zahorik, Customer satisfaction, customer retention, and market share, Journal of Retailing 69 (2) (1993) 193 – 215.

[5.1] N. Stroh-Maraun, D. Kaimann, J. Cox, more than skills: A novel matching proposal for multiplayer video games p 2.

[6] S. Karpinskyj, F. Zambetta, L. Cavedon, Video game personalization techniques: A comprehensive survey, Entertainment Computing 5 (4) (2014) 211–218.

[7] Howard Scott Warshaw, High Score: The Golden Era of Gaming: Boom & Bust timestamp 0:25 – 0:40 (2020)

[8] Wikipedia, Contents: History https://en.wikipedia.org/wiki/Artificial_intelligence_in_video_games accessed: 11th November 2021

[9] Becky Ann Heineman, High Score: The Golden Era of Gaming: Boom & Bust timestamp 7:39 – 7:50 (2020)

[10] Charles Martinet, High Score: The Golden Era of Gaming: Boom & Bust timestamp 11:13 – 11:55 (2020)

[11] Ernest Adams, Joris Dormans, Game Mechanics: Advanced Game Design p 26.

[12] https://www.gamedesigning.org/learn/game-design-principles/ Accessed: 11th November 2021

[13] Wikipedia https://en.wikipedia.org/wiki/Artificial_intelligence_in_video_games accessed: 11th November 2021

[14] Tomohiro Nishikado, High Score: The Golden Era of Gaming: Boom & Bust timestamp 10:45 – 10:55 (2020)

[15] Official Unreal Engine Documentation: https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/Blueprints/TechnicalGuide/NativizingBlueprints/

Accessed: 11th November 2021.

Appendix

Timetable

Information sources

Glossary

Term	Definition
UE	Unreal Engine
BP	Blueprints
Blueprints (in UE)	High level of abstraction Design system of coding style in editors.
Component	an object attached to another Object e.g., Inventory System is not IN the player is ON the player as Lego games
Callee	The object that receives an action from another object.
Caller	The initiator of the action to inform/trigger another object.
AI	Artificial Intelligence
Agnostic	Not Depended on the content e.g., no hardcoded
stochastic process	Non-Deterministic, more random believable
Deterministic	Same initial state same end results. Not truly random
VM	Virtual Machine
Actor	An Object spawned from class in the game world.
Shaders	A materials function component part that allows an object to be rendered and visualizes the color and shades of a surface and it gets executed in GPU. E.g., How an asset will look on render, how the object will respond to light.
Rig	Entire Skeleton System anatomy e.g., per bone linked

Animation	In a 2D world, either rig driven or rotoscoped in a 3D world driven by the rig (the bone parts can bend by simulating “muscles” via weight painting of the areas of 3d model manually or using a motion capture suit/software bundle setup).
texture	An Image that can be used in a material usually comes in a texture set which is a complex set of images called maps. Most of the time is UV wrapped
UV	2D coordinates of an image wrapped around a 3D model to create the “surface” look of it
material	A function applied in a 3D model to create the outer appearance of it (or part of it). Shader driven because if there is no light then no color (that has not to be absorbed) can bounce back to the camera’s “eye”.
NPC	Non-Player Character. AI-driven. But still; derived from the same class in Object Oriented Hierarchy.
Abstraction	A high-Level view of things from the final consumer perspective without knowing too much about its underlying mechanics but still able to use it.
rotoscope	Frame by Frame animating 2D images.
Actor reference	Spawned class == object
Real-time virtual production	Virtual production uses a suite of software tools to allow studios to combine live-action footage and computer graphics in real-time with animation. Contributors across multiple locations can create and render digital environments, while cast members are physically working on set.