Literature review

Konrad Jacek Marcjanowicz, U1968014

School of Computing and Engineering

Table of Content

Introduction

The demand for AR/ VR products is growing in the computer industry. The companies who were usually not associated with the possibilities of AR and VR are researching and developing new software and hardware. Newly rebranded Meta with the Metaverse platform to experience social activities within VR. Steam, the game company, made a high-end VR headset a couple of years, which was a huge success. As the VR/AR industry grows, there are new ways of producing VR software. Most popular development platforms already implemented support for VR and many online code bases and libraries have broaden their resources to support AR/VR. The development of such software might be challenging as the technology itself is relatively new, thus it is vital for the developers to choose appropriate tools to streamline its production.

Keywords: [VR, AR, game, games, development, virtual, reality, augmented, mobile, software, hardware, motion, tracking, graphics, headset, camera, glasses, lens]

Literature review

Software available to create VR/AR games

During the step where developer needs to choose which software to use, two major contestants display themselves on the market – Unity (unity.com, n.d.) and Unreal Engine (unrealengine.com, n. d.). Both of those platforms are free to use for personal and academic projects, rendering them accessible to anybody. Booth of them could be used to produce 2D, as well as, 3D environments that can be further extended into AR or VR. Unity and Unreal Engine are also amazingly easy and intuitive in usage compared to other types of platforms. They are what is called an RTE – real-time engine, this means that the game world can be always seen and inspected by the developer without the need of re-building of the project.

As those engines grew in popularity, they attracted new people who were talented programmers. Some of them would go forth and publish produced plugins for those platforms on their respective online marketplaces for free or at the cost. The database of those extensions also tackles VR/AR technologies. The Marketplace products can improve one's experience with programming VR/AR games by either simplifying the process, making the hardware integrate better with the development platform or tweaking the common bugs that might occur during the development process.

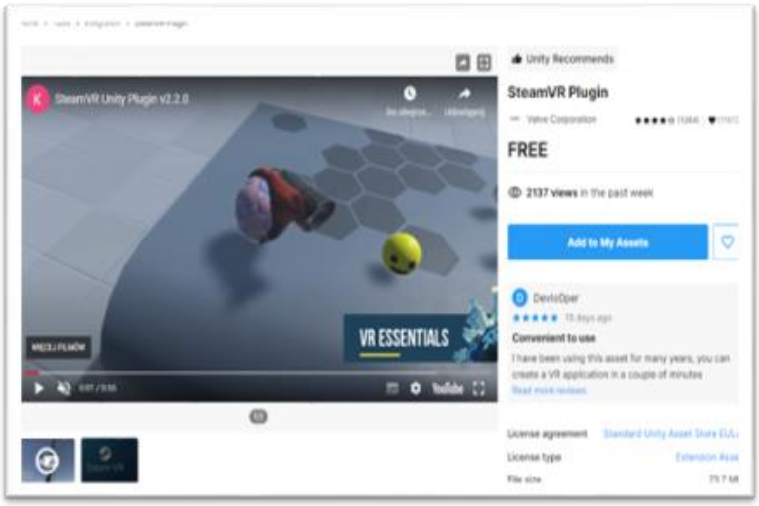

Figure 1. Plugin hosted on the Unity Marketplace for better incorporation of the Steam VR Hardware onto Unity platform.

As the RTE renders everything on the spot, the machine's hardware might struggle with displaying what has been put in the scene all at once. If that is the case, or if a developer prefers to do so, they can use a different approach – using libraries of code DirectX or OpenCV. Those libraries provide the user with all the functionality needed to produce the game – just without the clutter of already prepared game engine and the real-time. This solution allows the user to focus only on the functions that the product they are making requires. In addition, the developer can optimize the software better as they can view each specific part of the game and analyze the GPU usage. With the advancements and additions to debugging tools in the Visual Studio platform, as well as the support that Visual Studio offers for Direct X, looking for errors, debugging the code and understanding it can be done on frame and event basis, which may greatly improve the experience of using DirectX (visualstudio.microsoft.com, n.d.).

Hardware available to create VR/AR games

As VR aims to immerse people in its experience, a lot of VR headsets or full body motion trackers are produced to achieve that goal. They can range from standalone devices like Meta (Oculus) Quest to headsets that have all the necessary components to immerse the user but still need to be connected to a computer with VR software installed. The standalone devices are much easier to use and are generally more comfortable thanks to lack of media to which the gear needs to be connected, as well as, their weight. On top of that, they are usually cheaper to purchase. The shortcomings are though lower quality of a displayed image and lesser power of calculation, rendering most of the advanced games which require more resources unplayable.

Figure 2. Meta (Oculus) Quest 2 with two controllers

Tethered headsets on the other hand provide better graphics and more accurate tracking methods, like motion cameras and motion bands that can be wrapped around the angles or wrists. The resources to run the software and games are taken from the connected machine. One of the examples of those headsets is a high-end Valve Index. Valve index comes with two base stations. They are two laser projectors that sweep the play area in track of photonic sensors of the headset and controllers. That gives the user the highest possible range of poses and movement tracked by the VR gear without the inclusion of additional body trackers. (valvesoftware.com)

Figure 3. Valve index headset with two controllers and two base-stations

AR on the other hand does not need to immerse the user into its world. As the name suggests, it only “extends” the world that we see. Because of that a lot of different devices that can record and display an image can be used with the AR software. One of the most common devices is a smartphone. In recent years the smartphones became really advanced with powerful CPU’s, so they are able to run more demanding software and games. The best example of it would be using camera filters in social apps, like Instagram.

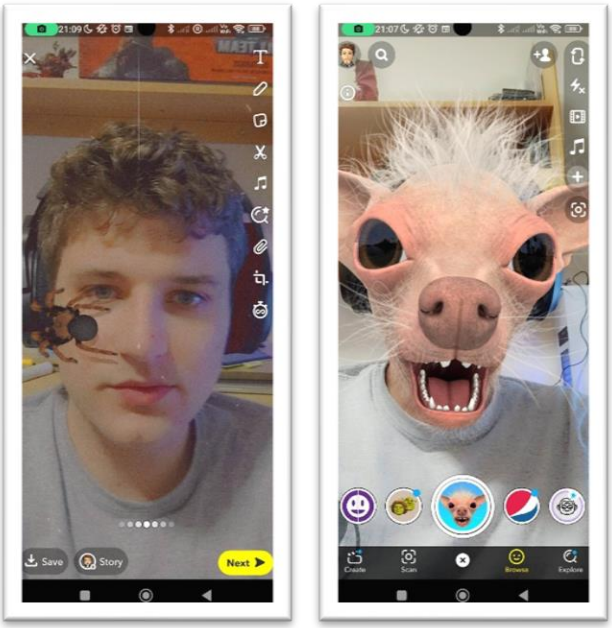

Figure 4 & 5. Camera filters with AR elements on popular social media app Snapchat

Design Report

Client's Game requirements

Customer asked to create a game for them that would fit in the busy shopping centre. The game will be displayed on all-in-one personal computer with Windows. The computer does have a camera above the screen. Customer wants the game to make use of it for the younger audience to control the game via any possible face movements or expressions. Even though the main audience for the software are children and young teenagers, the game should not exclude any potential player outside of the expected age range – the game should be accessible to everybody. The budget for the project is thin, so the customer would want to keep it as inexpensive as possible.

Target audience

The main target audience are children and young teenagers who are with their parents in the shopping centre, the secondary target audience is every person who comes to the shopping centre and wants to have some fun. This might span people of various ages – teenagers, young adults and elderly. The gameplay must also consider people who are impaired when it comes to movement and mental health.

The game and the game loop

The core mechanic of the game will be to "Fit it" – fit and object into a hole or groove and give that satisfying feeling of an object sitting perfectly when it is desired. Following the four main principles of game design (Jameson, 2017), the game would offer a challenge – the item would move left and right at the bottom of the screen, away from the desired place. The player would need to approximate if the item would fit into the hole moving to the hole in the straight line. Then the choice – the player decides when to stop the movement of the item by

blinking and try to fit it in place. After a successful fitment, the player could advance into a next level. The first levels might include:

- Put a ball into a tube
- Put a string through a needle hole
- Put an arm through a sleeve
- Get the boba tea bubble into a straw

The levels would get harder with every completion, and change the item and the hole that needs to be aligned. The time would also be measured for the player to give them the chance to improve their score on a leader board.

Chosen structure and work flow

To make this game, I will use one of the RTE engines – Unity. Thanks to the ease of use, as well as it supporting various 2D and 3D technologies, I will be able to quickly prepare the base for the project using one of the templates provided by the development platform. As the connection between Unity and OpenCV is supported, either via an extension on the Unity Marketplace or via exporting OpenCV code into a dynamic library, the game will use its face detection algorithms, as well as any functions necessary for real time calculations. As the customer needs to save money on a project I have chosen platforms that are both free commercially – Unity up to twenty thousand pounds per year and OpenCV is uniformly free to use by anyone.

Testing plan

This testing plan might be used during production of the game to test the particular logic behind the game to determine the success of development process. The table below is a

list of expected results, how the game should behave in the occurrence of events and the actual results from the testing phase.

Event in game	Expected result	Actual result
The face is being detected on the camera	The game switches from mode “on standby” and proceeded to display main menu and start button	
The player clicks the “Start” button	The game starts	
Player fails to align the object to its destined place.	Game restarts from level one, but the time already spent before failure is being counted towards the end.	
Player successes with aligning the object	Proceed to the next level	
Player finishes the game	Record the time and prompt for the name of the player, put the player in the leader board	
Face leaved the camera POV	Wait 5 seconds, then reset the game to “on standby” mode. Wait for another face	

Table of Figures

References

Unity Technologies. (n. d.). *Unity*. Unity. https://unity.com/

Epic Games, Inc.. (n. d.). *Unreal Engine*. Unreal Engine. https://www.unrealengine.com/

Microsoft. (n.d.). *DirectX Game Development*. visualstudio.microsoft.com. https://visualstudio.microsoft.com/vs/features/directx-game-dev/

Valve. (n.d.). *Valve Index.* www.valvesoftware.com. https://www.valvesoftware.com/en/index

Valve. (n.d.). *Base Stations 2.0*. www.valvesoftware.com. https://www.valvesoftware.com/en/index/base-stations

Jameson, D. (2017). *The Four Elements of Game Design: Part 1*. evantotuts+. https://gamedevelopment.tutsplus.com/articles/four-elements-of-game-design-1--cms-22720

Jameson, D. (2017). *The Four Elements of Game Design: Part 2*. evantotuts+. https://gamedevelopment.tutsplus.com/tutorials/four-elements-of-game-design-2--cms-25628