



Assignment of bachelor's thesis

Title:	Prototype of virtual reality game using various types of motion sensors
Student:	Oleksandr Khokhych
Supervisor:	Ing. Petr Pauš, Ph.D.
Study program:	Informatics
Branch / specialization:	Web and Software Engineering, specialization Computer Graphics
Department:	Department of Software Engineering
Validity:	until the end of summer semester 2022/2023

Instructions

Vytvořte prototyp hry ve virtuální realitě, která bude využívat různé typy senzorů pohybu pro sledování pohybu (např. senzory pro sledování dolních končetin). Prototyp zpracujte v prostředí Unreal Engine 4 (UE4).

1. Proveďte analýzu dostupných frameworků pro tvorbu virtuální reality v UE4.
2. Analyzujte senzory pro sledování pohybu ve virtuální realitě a jejich využití v UE4.
3. Analyzujte využití senzorů pro sledování pohybu v dostupném software pro virtuální realitu.
4. Na základě analýzy vyberte mechaniku, senzory a navrhnete prototyp hry.
5. Prototyp implementujte v UE4.
6. Proveďte vhodné testování.



**FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE**

Bachelor's thesis

Prototype of virtual reality game using various types of motion sensors

Oleksandr Khokhych

Department of Web and Software Engineering

Supervisor: Ing. Petr Pauš, Ph.D.

June 21, 2022

Acknowledgements

I would like to thank my supervisor for his help in writing this paper, my family and friends for their moral support.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No.121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60 (1) of the Act.

In Prague on June 21, 2022

.....

Czech Technical University in Prague

Faculty of Information Technology

© 2022 Oleksandr Khokhych. All rights reserved.

This thesis is school work as defined by Copyright Act of the Czech Republic. It has been submitted at Czech Technical University in Prague, Faculty of Information Technology. The thesis is protected by the Copyright Act and its usage without author's permission is prohibited (with exceptions defined by the Copyright Act).

Citation of this thesis

Khokhych, Oleksandr. *Prototype of virtual reality game using various types of motion sensors*. Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2022.

Abstrakt

Tato bakalářská práce se zabývá psaním VR hry na Unreal Engine 4 s použitím různých senzorů ke sledování pohybu. Hlavním cílem této práce je analyzovat technologie a metody sledovacích systémů používaných ve VR aplikacích za účelem vývoje vlastního prototypu imerzivní VR hry. Teoretická část práce popisuje základní principy vývoje VR aplikací v prostředí Unreal Engine 4, dále analyzuje stávající typy pohybových ovladačů, principy jejich fungování a oblasti použití. Praktická část je věnována volbě konceptu budoucí hry, input systému a také tvorbě a testování prvního prototypu. Výsledkem této práce jsou 2 hotové aplikace, s použitím ovladačů stop a bez nich, a také závěr o tom, který z těchto typů pohybu je vhodnější použít.

Klíčová slova virtuální realita, Unreal Engine 4, pohybové ovladače, sledovací senzory, VR hra.

Abstract

This bachelor's thesis is about writing a VR game on Unreal Engine 4 using various sensors to track motion. The main goal of this work is to analyze the technologies and methods of tracking systems used in VR applications in order to develop our own prototype of an immersive VR game. The theoretical

part of the work describes the basic principles of developing VR applications in the Unreal Engine 4 environment, as well as analyzes the existing types of motion controllers, their principles of operation and areas of application. The practical part is devoted to the choice of the concept of the future game, the input system, as well as the creation and testing of the first prototype. The result of this work are 2 ready-made applications, with and without the use of foot tracks, as well as a conclusion about which of these types of movement is more appropriate to use.

Keywords virtual reality, Unreal Engine 4, motion controllers, tracking sensors, VR game.

Contents

Introduction	1
Purpose	3
1 VR with Unreal Engine 4	5
1.1 What is Unreal Engine	5
1.2 VR experience in UnrealEngine4	6
1.3 Unreal Engine 4 VR Template	7
2 VR Motion controllers	9
2.1 Locomotion System (LS)	9
2.2 Position Tracking System	11
2.2.1 Acoustic	12
2.2.2 Magnetic	12
2.2.3 Inertial	13
2.2.4 Optical	14
2.3 VR Motion Controller Types	16
2.3.1 Head Mounted Display	16
2.3.2 Hands Motion Controllers	20
2.3.3 Legs Motion Controllers	23
2.3.4 Motion Capture Controllers System	25
3 Analysis of existing VR applications	27
3.1 Richie's Plank Experience	28
3.2 Super Hot	29
3.3 The Unspoken VR	30
3.4 Stride	31
3.5 VR Chat	32
4 Game design concept	33

4.1	Goal	33
4.1.1	Criteria for VR app	33
4.2	Main Idea	34
4.2.1	Inspiration - Subway	34
4.3	Game concept	35
4.4	Requirements	37
4.4.1	Functional	37
4.4.2	Non-Functional	38
4.5	Chosen controllers and input system	39
4.5.1	VR helmet — Oculus Quest 2.	39
4.5.2	VR motion controllers — Oculus Touch	40
4.5.3	Additional sensors for tracking legs — Kat loco S	41
4.6	Sequence diagram	42
5	Implementation	43
5.1	Plugins	43
5.2	Base world mechanics	44
5.2.1	Auto run and increase game difficulty	44
5.2.2	Procedural generation	44
5.3	Controllers input tracking	45
5.3.1	Acceleration	46
5.3.2	Dodging	46
5.3.3	Jumps	47
5.3.4	Hits reaction	48
6	Testing	49
6.1	Mechanic testing results	49
6.2	Gameplay testing results	51
	Conclusion	53
	Bibliography	55
	A Acronyms	59
	B Contents of enclosed CD	61

List of Figures

2.1	DoF principle of work[1]	10
2.2	Optical tracking solutions[2]	15
2.3	Stereoscopic image basic principle of work[3]	16
2.4	Example of a VR headset for mobile phones[4]	17
2.5	Example of a stationary VR headset[5]	18
2.6	Example of a standalone VR headset[6]	19
2.7	Example of motion controllers[7]	20
2.8	Example of a haptic gloves[8]	21
2.9	Example of an optical hand motion tracker[9]	22
2.10	Example of foot tracking sensors[10]	23
2.11	Example of a non-physical VR locomotion controller[10]	24
2.12	Example of full body trackers[11]	25
2.13	Example of a mo-cap suit[12]	26
3.1	Richie’s Plank Experience game preview[13]	28
3.2	Super Hot game preview[14]	29
3.3	The Unspoken VR game preview[15]	30
3.4	Stride game preview[16]	31
3.5	VR Chat game preview[17]	32
4.1	Subway Surfers game preview[18]	34
4.2	Example of a generated map	35
4.3	Example of a chunk with obstacles	35
4.4	Screenshot of gameplay during the crouch	36
4.5	Screenshot of gameplay during the jump	36
4.6	Demonstration of how character collision works	39
4.7	Demonstration of the principle of jump activation	40
4.8	Demonstration of the principle of acceleration activation	41
4.9	Sequence diagram	42
5.1	List of plugins used to work with VR controllers	43

5.2	The logic of auto-running and increasing the game's difficulty. . . .	44
5.3	Procedural map generation from ready-made chunks.	44
5.4	State machine for logic tracking controllers.	45
5.5	Plugin for collecting data from foot controllers	45
5.6	Acceleration calculations based on data from the foot controllers .	46
5.7	Implementation of a dynamic player collision	46
5.8	Calculating the velocity of the hand motion controllers	47
5.9	Implementation of upward jump tracking.	47
5.10	Implementation of sideways jump tracking.	47
5.11	Player collision handling	48
5.12	Tactile effect implementation	48

List of Tables

6.1 Overall testing results	50
---------------------------------------	----

Introduction

Virtual reality (VR) is a world simulated by computer technology in which the user can immerse themselves through the use of special sensory devices. Their impact on human receptors, complemented by the imitation of interaction with the world, creates the impression of a real presence in the VR world. To achieve all of this, in response to the user's actions, the VR system processes and outputs information in real time.

Unlike Augmented Reality (AR), where the image only supplements the real picture, VR will show you a fully simulated world. This opens up an infinite number of possibilities for improvisation, creation, and representation of reality as a whole. VR technology has come a long way, from its first inception in the 1960s, to today's VR helmets. A new wave of interest in VR began thanks to the company Oculus and the prototype Oculus Rift glasses presented in 2012. Since then, the interest in this technology has only grown from year to year, which makes this platform more and more promising and attractive for investment. Many companies choose VR to create and promote their own products or use integration into existing applications and market-places. It is also widely used in game development, film-making, education, construction, and many other industries.

Not everyone has a VR headset at this moment, but looking at the growing trends of active users, it's only a matter of time before we all move into the new reality. There is still time for crazy experiments in the hope of taking your place in the market of the future. That's exactly what I'm going to do with my work.

Purpose

The main aim of this paper is to analyse the technologies and locomotion techniques used in VR applications in order to develop our own immersive VR game prototype on the Unreal Engine 4. The main difference of this prototype from other VR games is the unique developed kind of physical locomotion using different kinds of motion trackers. The most difficult task in creating this solution was to meet all functional and non-functional requirements. The result is a finished prototype of the game, tested for usability and comfort of the implemented locomotion.

VR with Unreal Engine 4

1.1 What is Unreal Engine

Unreal Engine is a game engine created by Epic Games in 1998. It was first used to create the first person shooter game, Unreal, and has since been used to create a numerous large and complex games (including Fortnite, Mortal Combat, BioShock, Outlast and many others). Written in C++ , Unreal Engine has a high degree of mobility, which makes it cross-platform.

Since its creation, besides being used to create games of various genres for all platforms, it has also been widely used in the film industry. Its graphical capabilities come close to photo-realism; the utilization of post-processing and creation of special effects in real time, made it possible to speed up and reduce the cost of filming in many ways.

The engine also contains a very extensive and convenient toolkit for developers. In addition to advanced functionality for working with animations, simulations, sounds, and particle effects, it provides two alternative ways of writing code:

- **The C++ programming language**, familiar to many programmers, allows you to use the full potential of the engine. The developers also offer free access to the source code of the engine stored on their official GitHub repository. This allows you to look into the engine's code and build/extend upon it to your liking. You can also write your own plugins and integrate them.
- **Visual programming** that does not require the user to have programming experience to write their first game. Based on its own scripting system called blueprints, it allows you to easily interact with engine components and create gameplay elements using a visual interface. It is a very flexible and powerful system that allows designers to use the concept and almost the full potential of programming.

1.2 VR experience in UnrealEngine4

VR support has been available since engine version four, in 2012. The first supported device was the Oculus Rift DK1. Today, thanks to the support of OpenXR standard, the engine already works with all major devices.

OpenXR is a free standard that aims to standardize the development of virtual and augmented reality applications, increasing hardware and software compatibility[19]. It is supported by almost all major hardware, platform and engine manufacturers in the VR industry. Since the Unreal Engine version 4.27, the developers have completely switched to the OpenXR and will only support it in the future[20]. Now it is greatly simplify the process of developing game projects for different VR platforms. Developers don't need to write separate code to support a particular device. Thanks to emulating any controller input mapping by using OpenXR interaction profiles, it no longer matters which VR helmet or controllers you will use[21]. In addition to basic functionality such as controller position tracking, input system control, etc., the OpenXR plugin can be extended within the engine by many different plugins. For example:

- *OpenXRHandTracking* — Provides basic functionality for real-time hand recognition without the involvement of controllers. Data from sensors about the position of bones and joints are returned to the user in the form of structures with information about each element for further processing. It is often used to recognize certain gestures used by the system. The processed poses are sent and processed as one of the input sources.
- *OpenXREyeTracker* — Using a special sensor device that tracks the eyes and accurately displays what the user is looking at. The main purpose of this extension is to provide a point of focus for the gaze. Some applications use this method as one way of interacting with the virtual environment.
- *Oculus OpenXR* — The plugin used to support Oculus headsets and features. It is provides additional functionality for more advanced work with Oculus devices. In the future, I will use it for the implementation part of my work.

The following platforms have released OpenXR runtimes and are currently supported in Unreal Engine:

- Windows Mixed Reality,
- HoloLens 2,
- Oculus,
- SteamVR.

1.3 Unreal Engine 4 VR Template

If you are looking for a starting point for developing a project using OpenXR, the engine developers offer a ready-made VR template with pre-built base classes. The following objects define the rules of the game experience and how it is configured:

- **VRPawn** — In Unreal Engine, a Pawn is the physical representation of the user and defines how the user interacts with the virtual world. In the VR Template, the Pawn contains the logic for input events from the motion controllers.
- **VRGameMode** — The Game Mode object defines the rules of the game. In it, you can configure which pawn you want to use, or select a certain class of motion controllers. It can also contain the basic logic of the game world's behavior.
- **Player Start** — The Player Start defines where the Pawn is spawned in the virtual world. For VR experiences, the Player Start's origin is the tracking origin in the virtual world.

In addition to the base classes, the template contains an implementation of many useful features. It includes encapsulated logic for:

- **Teleport Locomotion** — This is the first basic type of movement in virtual space. It is done by selecting an available point on the map and teleporting the player directly there.
- **Snap Rotation** — This is the second basic type of movement in virtual space. Provides smooth rotation of the player.
- **Grab Component** — A component that provides basic functionality for interacting with game objects. You can grab them and use the extra actions they provide using the controller buttons.
- **VR Spectator Camera** — With Spectator Screen Mode, others can view the VR experience while someone uses the head-mounted display (HMD).
- **Menu** — Implementation of a simple user interface in VR.

The VR platforms currently supported by VR Template include:

- Oculus: Quest 1, Quest 2, Rift S,
- Valve Index,
- HTC Vive,
- Windows Mixed Reality.

VR Motion controllers

VR Motion controllers are hardware accessories that allow users to perform actions (locomotion) in VR by using various sensors to track movements and input data. The number of controllers used, their type, compatibility and capabilities directly affect the VR gaming experience.

2.1 Locomotion System (LS)

VR locomotion is a technology that allows you to move from one place to another (locomotion) in a VR environment. It is one of the main design features of any VR application, because ensuring comfortable and efficient movement directly affects the degree of immersion and the resulting game experience. Moving is done when we move in the virtual world, or the virtual world moves around us. The movement itself can be accomplished through one of two types of locomotion[22]:

- **Physical locomotion** is when the movement in the virtual world is directly controlled by the movement of controllers in the physical world. By using a position-tracking system, with data about the position of the controllers, the virtual world repeats our movements imitating full presence on stage. Thus we can not only see our hands, body or feet in the virtual world, but also interact with environment.
- **Artificial locomotion** is when the movement in the virtual world does not directly correspond to the physical movement of the controllers. All movements are performed by receiving signals directly from the controllers themselves by pressing various control elements (buttons, trigger, sticks, sensors...). The most common use of artificial movement is to allow people to move around in a virtual environment that is larger than their physical gaming space.

2. VR MOTION CONTROLLERS

There are many familiar types of controllers for artificial locomotion, such as computer mouse, keyboard, gamepad, etc. However, despite their prevalence, they are less and less often used as VR controllers. The reason for this is the unnaturalness in their use. The control system in VR should be as close as possible to our natural interaction with the real world. Most of this we can achieve with a well-implemented tracking system. By synchronizing movements with our virtual avatar, we can increase the level of human self-perception in VR. The more tracking capabilities we have, the more immersive and interactive experiences we can achieve. But immersion is not only a matter of tracking capabilities and accuracy, but also of its stable operation.

Before explaining how the tracking system works, it is necessary to introduce the concept of degrees of freedom.

Degrees of freedom (DoF) - the number of different basic ways of moving a rigid body inside space that do not affect each other[23]. There are only 6 degrees of freedom in total, and we can divide them into two different types:

- **Translations.** The body can move freely along three DoF: up, right, and forward.
- **Rotations.** The body can rotate with three DoF: pitch, yaw, and roll.

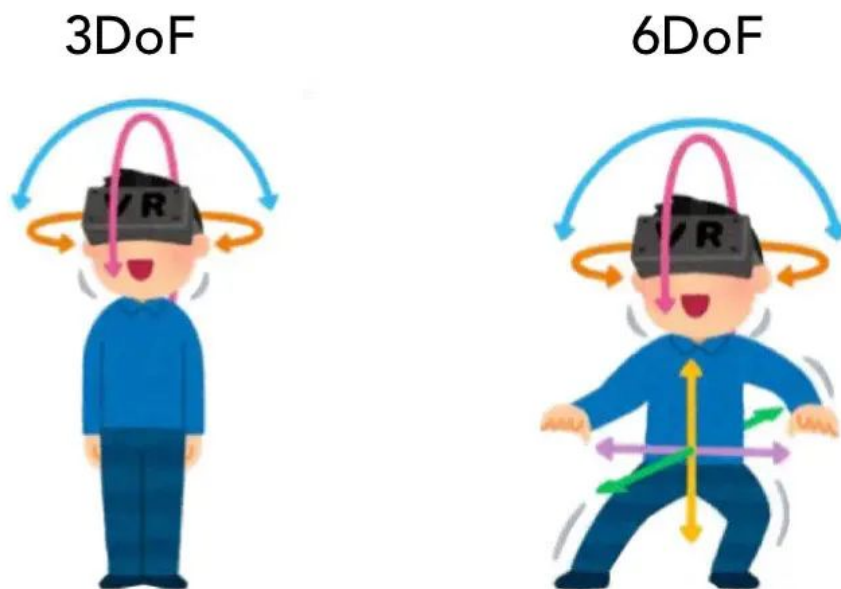


Figure 2.1: DoF principle of work[1]

2.2 Position Tracking System

Position tracking is a combination of hardware and software algorithms that can determine the absolute position of an object. It is used in many virtual reality technologies to determine the user's location in the real world. For full immersion, it is important that we can accurately track how objects, such as VR helmets or controllers, move in real life to represent them in virtual reality. Through a combination of location tracking (controller translations) and orientation tracking (controller rotations), it becomes possible to measure and report on all 6 degrees of freedom of the real world. The data from the sensors is directly processed by the app to show the response to the user's actions.

Examples of such technologies include:[24]

- **Head and eye tracking.** This tracking system is fundamental to the operation of many visual displays. Using data about head position as well as pupil size and rotation, we can accurately determine the direction of gaze and focus point for rendering a particular image and user interaction with its elements. Systems of this plan are widely used in VR and AR where helmet position and user's pupil position are an important source of information and input.
- **Hand and arm tracking.** This tracking system is used to create remote control techniques and haptic interfaces. Its applications are tele-robots, dynamic hand simulations in both real and virtual worlds. Using data about the position of hands and fingers, it is possible to reproduce accurate manipulation of remote or virtual objects and to recognize various gestural commands.
- **Body tracking.** The main application of this tracking system is locomotion. By reading the position of the whole body in space, we are able to process and visualize these movements. This system is used extensively in motion capture for film-making, making it much easier to rig and animate characters. And also its use is one of the main condition for full immersion in virtual reality.

There are many ways to achieve positional tracking, with pros and cons for each solution. These systems vary in such aspects as accuracy, resolution, sampling rate, latency, range, workspace, cost, load, convenience, susceptibility to obscuration, ease of calibration, number of simultaneous measurements, and orientation versus position tracking and can be divided according to their technical execution into the following groups[25]:

2.2.1 Acoustic

Acoustic trackers use high frequency sound waves to determine the position and orientation of an object in space. Typically, multiple transmitters are distributed in the monitored environment and multiple receivers (microphones) are placed on the monitored objects.

With at least 3 transmitters and at least 3 receivers, this method allows to calculate all 6 DoF. The principle of operation of this method is similar to the work of echolocation in animals.

There are two types of positioning using sound waves:

- measurement of the time of flight of a sound wave from a transmitter to receivers,
- calculation of the phase difference of a sinusoidal sound wave during reception and transmission.

Pros:

- accuracy in calculating the position and rotation of the controller,
- the sensors are small and lightweight, allowing more flexibility in implementation,
- the devices are cheap and easy to manufacture.

Cons:

- slow position update rate caused by the low speed of sound in air,
- problems of environment in which the monitoring takes place (temperature, pressure, humidity, extraneous sound signals).

2.2.2 Magnetic

Magnetic tracking is based on measuring the intensity of a magnetic field in different directions. The source of generating an alternating or static electromagnetic field, as well as the reference point of the coordinate system is the base station. When the measurement point moves relative to its base, the strength of the magnetic field changes in inverse proportion to its distance. Thanks to this, we can accurately determine the location of the controller. If the measuring point rotates, the distribution of the magnetic field changes along different axes, which makes it possible to determine its orientation.

Pros:

- provides accuracy without any hardware requirements and a low total cost,
- good measurement accuracy.

Cons:

- user needs to be close to base emitter,
- tracking worsens near metals or objects that interfere with the electromagnetic field,
- requires constant calibration due to frequent errors and inaccuracies.

2.2.3 Inertial

Inertial tracking is a combination of several sensors collectively providing information about the position and rotation of the controller. The system consists of:

- **Accelerometers** — sensors measuring linear acceleration. Since the time derivative of position is velocity and the speed derivative is acceleration, the accelerometer output can be reintegrated to find velocity and then reintegrated again to find position relative to some starting point.
- **Gyroscopes** — sensors that measure angular velocity. The angular velocity can also be integrated to find the angular position relative to the starting point.

Modern inertial systems allow you to track orientation (roll, pitch, yaw) in space with a high update rate and minimal delay. However, you cannot rely completely on inertial tracking to determine position. Integration and double integration cause drift to increase quadratically with time, resulting in large errors. Often this system is combined with any other tracking location to correct for position tracking.

Pros:

- good reads fast movements especially in combination with other sensors,
- extremely fast update rate of sensor data.

Cons:

- due to the high cumulative error, it is possible to obtain accurate data for all 6 DoF only in combination with other types of sensors.

2.2.4 Optical

Optical methods are a combination of computer vision algorithms and tracking devices to determine the position and orientation of the controller. The sensors used are visible or infrared cameras, stereo cameras, and depth cameras calibrated to determine the distance to an object and its position in space. Optical systems are very reliable but require good illumination without overlap to obtain correct data. Depending on the choice of reference system, there are two approaches for position tracking:

Outside-in

The outside-in approach implies the presence of static external cameras that determine the position of a moving object by characteristic points. The accuracy of this approach depends on the number of external cameras and the area they monitor.

Pros:

- low delay relative to the inside-out method,
- more accurate readings depending on the number of cameras placed.

Cons:

- the playing space is only within view of the cameras,
- the tracked object must always be in the field of view of the cameras.

Inside-out

The inside-out approach assumes the presence of an optical sensor on a moving object, thanks to which it is possible to track movement relative to fixed points in the surrounding space. This method does not require the presence of additional cameras in addition to those built into the controller, which makes it convenient to use for any room.

Pros:

- allows you to customize the playing space for any room,
- adapts to new tracking environment conditions.

Cons:

- higher latency due to computation on the device itself,
- smaller camera field of view than the opposite method.

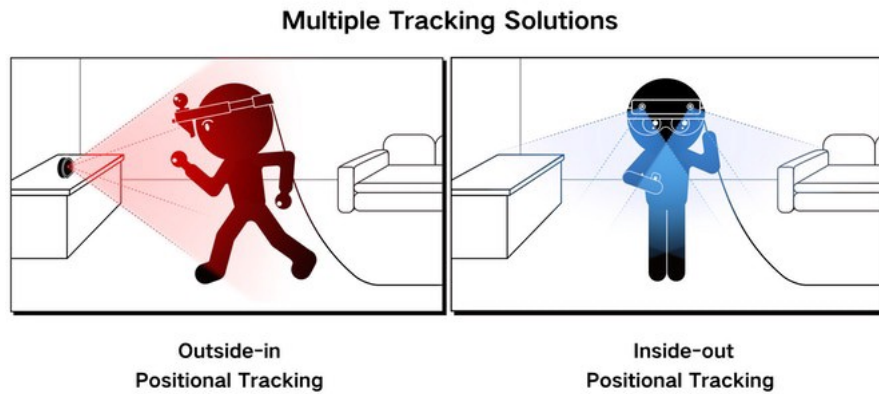


Figure 2.2: Optical tracking solutions[2]

Depending on the presence of special optical markers are distinguished separately:

Marker less

Marker less tracking is usually based on complex algorithms using cameras with depth sensors. Using the depth map received from the cameras, it is possible to find the volume of the object we are looking for or calculate the camera's offset relative to the initial position.

- The first case implies a pre-loaded model, which the system looks for in the image from the cameras. based on this, the position of the controller in space is determined.
- The second case works on the principle of calculating the distance to visible objects using the depth camera. Based on the data obtained, it is possible to detect the tracked object.

Marker based

Tracking using markers involves a pre-defined marker image or infrared signal frequency that can be tracked even with a single camera. This type of tracking is possible only within the line of sight of the marker.

- Infrared sources (both active and passive).
- Visible markers like QR codes.

2.3 VR Motion Controller Types

The following will describe the categories of different types of controllers, as well as the difference between the representatives of each group.

2.3.1 Head Mounted Display

The most common controllers used for immersion in VR are specialized helmets/glasses. The display in front of the user's eyes displays stereoscopic image.

Stereoscopy is a technology that creates the illusion of three-dimensional depth using a pair of two-dimensional images. By rendering two 2D images of the virtual world from different angles for the left and right eyes respectively, stereoscopy tricks the brain into giving us the illusion of volume.

How to create stereoscopic 3D images

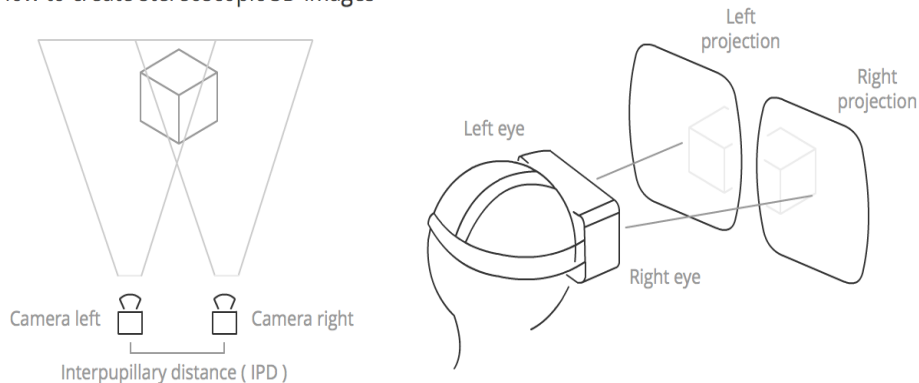


Figure 2.3: Stereoscopic image basic principle of work[3]

Usually, VR helmets have at least 3 DoF. The gyroscope and accelerometer attached to the body track head rotation and transmit the data to the computing system, which changes the image on the display depending on the readings of the sensors. As a result, the user is able to "look around" inside the VR and feel in it as in the real world. More advanced helmets have six degrees of freedom, which allows in addition to rotation, also track the location of the helmet in space. Now the player can not only turn his head in the virtual world, but also move through the virtual space by shifting the position of the helmet in the real world.

All VR helmets are divided into three types:

For mobile phones

In this type, all calculations are carried out on a mobile device attached to a helmet with lenses. The phone's display acts as a screen, and the helmet's position in space is determined by its accelerometers and gyroscopes. Control is carried out using the buttons on the mobile device itself, additional buttons on the helmet or a gamepad connected to it. The result and quality obtained when using helmets of this type to a greater extent depends not on the helmet, which is only a body, but on the device that is inside it.

Pros:

- cheapness and ease of use. If you have a mobile phone, your first VR helmet can be assembled from cardboard,
- it's a portable solution that doesn't restrict movement and doesn't require a predefined game room.

Cons:

- supports only 3 DoF,
- requires a smartphone to work,
- the picture quality and performance of the whole system depends on the computing power of the mobile device.

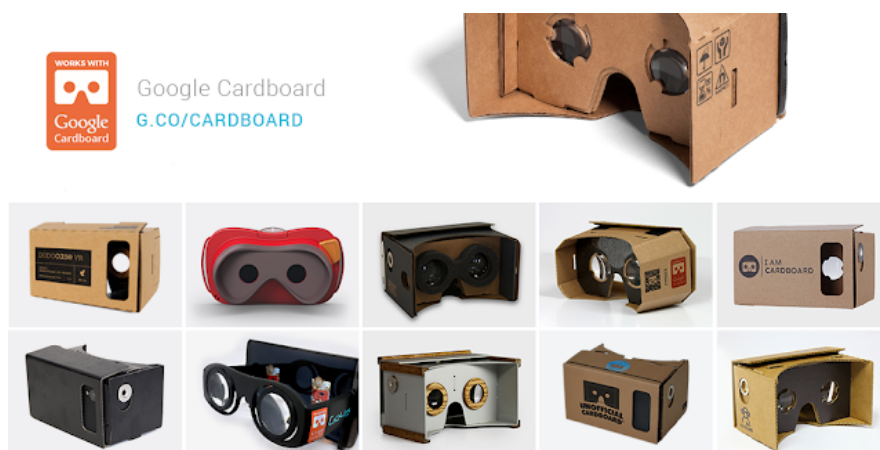


Figure 2.4: Example of a VR headset for mobile phones[4]

Stationary

This type of VR glasses already has its own display installed in the housing. The headset is connected directly to the computer through a cable and broadcasts the calculated image received from it. Due to the calculations on the computer, the helmet is easy to use and does not require additional maintenance on the part of the user. Third-party tracking systems are often used to track the position of the helmet. One example of these is tracking base stations located in the gaming area.

Pros:

- support for all 6 DoF,
- the solution for the most performance-demanding applications,
- the helmet does not require recharging.

Cons:

- require stable cable connection and limited in movement by its length,
- requirement in additional devices to track the position of the helmet and controllers.



Figure 2.5: Example of a stationary VR headset[5]

Standalone

The most advanced version of the VR helmet has its own screen and hardware with all the necessary software. In essence, they are in many ways an improved version of mobile helmets by integrating the hardware inside the body and adding additional sensors to track its position. These devices have their own operating system, and also make all the calculations for the position of the helmet and its associated controllers themselves.

Pros:

- do not require any additional equipment,
- support for all 6 DoF,
- have no travel restrictions,
- have the ability to connect to a computer via cable and further use as a stationary version.

Cons:

- the need to charge the battery,
- mobile hardware is still far from the performance of modern computers. This limits the list of available applications on this platform.



Figure 2.6: Example of a standalone VR headset[6]

2.3.2 Hands Motion Controllers

This type of controllers, by their nature, are intermediary devices between our real hands and their graphic projection in VR. They often use motion tracking technology, gesture interfaces and finger tracking, and some models even allow for tactile sensations. Thanks to this we can not only see our virtual representation of hands, but also interact with the virtual space as realistically as possible.

Tracked Motion Controller

Controllers of this type, thanks to the physical body, have a certain position in space, which allows you to interact precisely with digital objects. The presence of additional buttons and triggers on the controller makes it possible to expand the range of possibilities for these interactions. Sticks allow you to move and rotate your character in the virtual world without using any additional sensors. Thanks to their ergonomics and ease of use, they become an integral part of any VR experience.

Pros:

- have additional buttons for enhanced interaction,
- are cheap and easy to use.

Cons:

- have a weak tactile effect and naturalness of use compared to other hand controllers.



Figure 2.7: Example of motion controllers[7]

Haptic Gloves

This is the most advanced hand controller solution available today, allowing you not only to track the position of each finger in space, but also to influence the motor skills of your hands by means of servo drives. This allows us to simulate grasping virtual objects by feeling their volume in reality. An additional tactile effect is achieved by the vibrations created by the vibrators. All this allows you to imitate tactile sensations, taking the game process to a whole new level.

Pros:

- the best haptic controllers at the moment,
- allows to track not only the position of hands, but also each finger.

Cons:

- very expensive,
- not the most accurate way to track the position of fingers and hands in space,
- not the most user-friendly controllers.

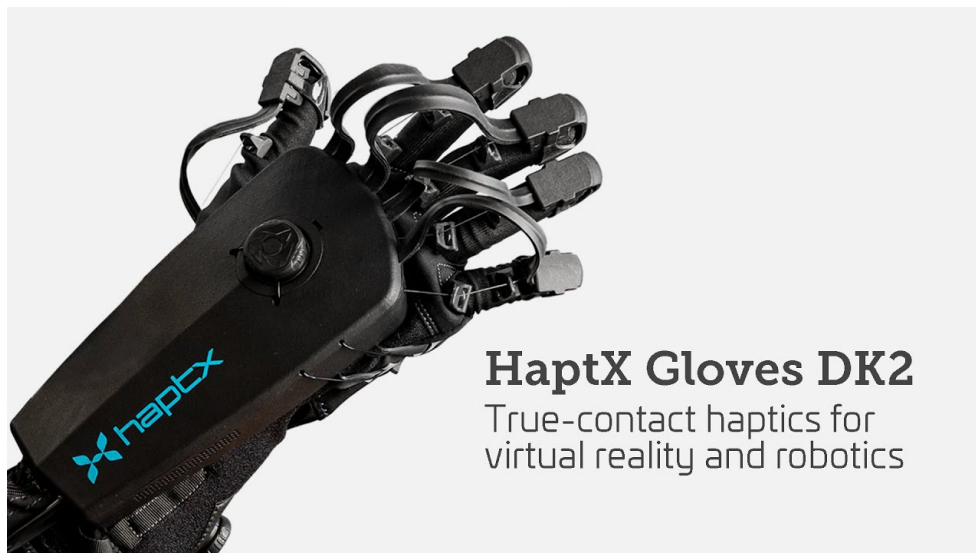


Figure 2.8: Example of a haptic gloves[8]

Optical Hands Tracking

This type of controller differs from the previous two in the nature of its use. It does not require physical objects in the hand or additional motors on them. All interaction takes place in the air without touching or pressing extraneous buttons. Thanks to optical sensor tracking, we can determine the position and posture of the hand in space. Based on this, it is not only possible to reliably visualize the hands in VR, but also to read and process certain gestures. Using ultrasonic wave technology of varying frequencies, we can transmit tactile sensations to the surface of the user's hands, simulating touch.

Pros:

- provides a good tactile system,
- provides the most accurate hand and finger tracking,
- natural interaction without the presence of extra objects in hand.

Cons:

- very expensive,
- hand tracking and haptic only within the camera's field of view,
- has some limitations with haptics in air.



Figure 2.9: Example of an optical hand motion tracker[9]

2.3.3 Legs Motion Controllers

Leg controllers are not as common in use, but their presence directly affects the gaming experience. Using them, you can not only accurately produce movements in VR (like jumps, sprint...), but also get rid of motion sickness and disorders of the vestibular system.

Tracking Sensors

These are already familiar motion sensors that track only 3 degrees of freedom. Attached to the legs, belt, and working in pairs, they provide information about the direction of the player's torso and the movement of his legs in the real world. Possessing only rotation data, they allow to recognize some patterns of behavior and use this information to carry out various types of movements in virtual reality.

Pros:

- a cheap solution for realizing locomotion using legs in virtual reality,
- it has pre-configured templates for recognizing certain types of movements.

Cons:

- it has only 3 degrees of freedom, which makes it impossible to visualize the position of the legs in virtual space.



Figure 2.10: Example of foot tracking sensors[10]

Input controllers

This type of controller has no motion tracking sensors and works by sending input commands. In essence, they are a kind of interface for interacting with the virtual application by means of leg movements. By fixating their position in space by sitting down or strapping in, the player is able to move their feet freely in space. Whether it is rotating the roller on the soles of the controllers' boots, deflecting the joystick under your feet or sliding on the touch surface - all these and many other features make the virtual experience more dynamic and interactive.

Pros:

- a wide variety of controllers in different price categories,
- do not require additional computing stations to operate the controllers.

Cons:

- have no degrees of freedom,
- restrict the player's movement by chaining him to one position,
- have very limited functionality.



Figure 2.11: Example of a non-physical VR locomotion controller[10]

2.3.4 Motion Capture Controllers System

The Full Body Tracking system is used to track all of the user's movements as accurately as possible. The combination of a large number of sensors located all over the body and working in parallel allows us to transfer all the capabilities of our locomotion for realistic animation of the virtual avatar. This system finds wide application in cinematography and social applications with live communication, where its potential is maximized.

Full body Tracker

This type of controllers allows you to track the position of any object in space, to which it will be attached. Supporting the tracking of all 6 degrees of freedom, it is a simple and convenient addition to any existing tracking system. For example, tracking the legs and torso with their complement of inverse skeletal kinematics, allows you to determine the position of the entire body quite accurately.

Pros:

- compatibility with other tracking systems,
- accurate tracking of all six DoF.

Cons:

- this type of controllers requires the use of additional equipment for their tracking.

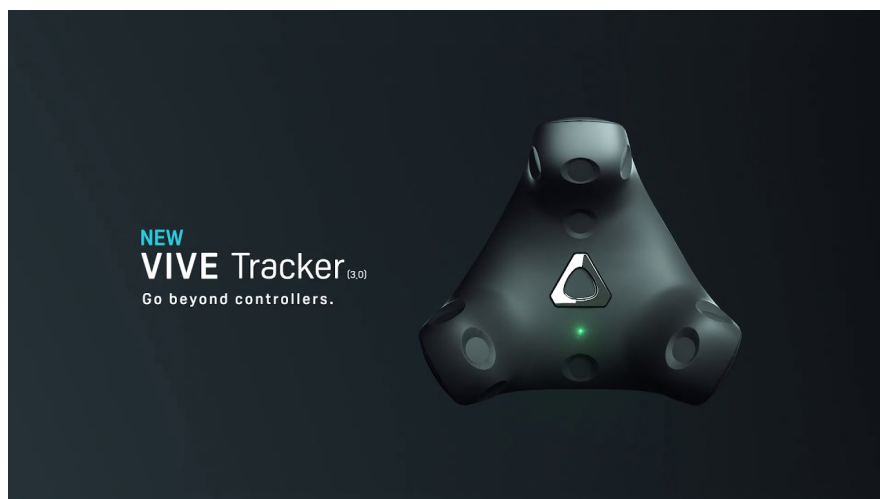


Figure 2.12: Example of full body trackers[11]

2. VR MOTION CONTROLLERS

Full body Suit

In essence, it is not a separate motion controller, but a technical and software combination combined into one suit. It is the most advanced full body motion capture solution currently available. It is very often used in combination with facial motion capture as well as hand motion capture. The range of applications of this suit is very wide, but in spite of this, its main task remains capture of human movements in the real world, transferring them into a digital format for further post-processing.

Pros:

- the most precise type of full-body motion capture,
- it has a very high range of applications,
- it is a monolithic design that does not require individual sensors to be connected to each other,
- some solutions have built-in support for tactile effects.

Cons:

- very expensive,
- requires additional devices to track the sensors in the suit.

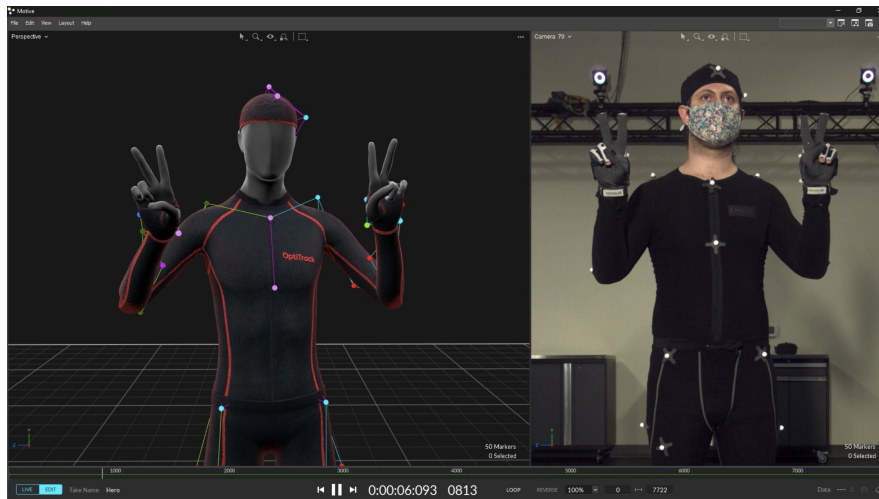


Figure 2.13: Example of a mo-cap suit[12]

Analysis of existing VR applications

Gaming and entertainment is currently the best known and widest use of VR. It includes games as well as movies, virtual tourism and attending various events. The number of out-of-the-box products is growing along with the sales volume of VR headsets year after year, attracting more and more users. My task in this chapter is to analyse popular gaming applications and the types of tracking systems they use.

The main criteria in my search and selection were the following points:

- strong immersion effect,
- comfortable and dynamic gameplay,
- interesting implementation of locomotion in virtual reality,
- interesting implementation of input system,
- extensive support for various tracking systems.

Below I will give an example of the projects I am most interested in, which have some features from the list above:

- Richie's Plank Experience,
- Super Hot,
- The Unspoken VR,
- Stride,
- VR Chat.

3.1 Richie's Plank Experience

Main idea

This app is essentially a VR experiment, testing how seriously the player takes virtual reality. The game forces the user, immersed in virtual reality, to experience the fear of being on top of an 80-story building. The conditions, which are out of the realm of everyday life, test the naturalness of reactions to the virtual world. The player moving on a thin board leading from the edge of the roof directly over the precipice, feels the fear of heights and serious danger, and begins to involuntarily believe in the veracity of what is happening. The strength of the effect sometimes depends greatly on whether the user has had previous experience with virtual reality.

Required controllers

For the full functionality of the idea, this application requires only the presence of a vr helmet with tracking 6 degrees of freedom. The displacement of the helmet in space, directly affects the position of the player on the board. If the player takes a wrong step and moves away from the virtual board by a certain distance, his avatar will start to fall down quickly.

Recommendation

The presence of position tracking sensors on the legs with 6 degrees of freedom, would increase the effect of immersion, due to the reliable visualization of the position of the player's feet on the board.



Figure 3.1: Richie's Plank Experience game preview[13]

3.2 Super Hot

Main idea

This game is a dynamic first-person shooter in virtual reality. Its main distinguishing feature from other products of this category is its unique mechanics of interaction with the world. Time in the game starts to run only when the player moves. This mechanic is ideal for virtual reality, where the player has a greater potential for movement than in the PC version. An additional plus of this game is the ability to interact with various objects in the environment, making the virtual reality more immersive. Dodging flying bullets, shooting enemies with guns and throwing various interactive objects at them, the player feels like a real hero of the Matrix movie.

Required controllers

To test yourself as Neo, the player will need a virtual reality helmet with 6 DoF. The tracking system will repeat head movements in the virtual world, as well as being a body and a target for enemies. Interaction with game objects is made with hand motion controllers also having support for all six DoF and a basic set of buttons for shooting and grabbing.

Recommendation

Using foot sensors would improve immersive effect by adding interactivity involving them. The ability to kick or push an opponent would be a very nice addition to the rest of the game's mechanics.



Figure 3.2: Super Hot game preview[14]

3.3 The Unspoken VR

Main idea

This is a multiplayer game, the main actions of which are magical duels between players. Casting a variety of spells with the movement of their hands in the air and dodging enemy ones, teleporting along the platforms, the task of the players is to lower the enemy's health value to zero. In this they are helped by a large list of movements, triggering a variety of magical abilities, traps, attacking and defensive spells. The process of creating and casting spells with hand movements is native and immersive enough for VR to make this a game worth trying out.

Required controllers

This game requires the same basic type of controllers, namely a helmet and two motion controllers. In this case, the position of the helmet does not play a particularly large role, since all movement is carried out through teleportation, but instead, tracking of hand controllers will take a special place. It is based on their position and movement that the desired spell is triggered.

Recommendation

The use of optical tracking in conjunction with the AI hand recognition system would allow casting spells based not only on the position of the hands, but also on special gestures and the position of the fingers. This will make the core mechanics even more convenient and immersive.



Figure 3.3: The Unspoken VR game preview[15]

3.4 Stride

Main idea

Stride is an action parkour game in which the player is always on the move. Having a large number of different mechanics and modes available, it allows you to try yourself as a real parkour sprinter. In the course of gameplay runs through an endless procedural generated level or skyscraper rooftops, the player will need to use the potential of his VR headset and controllers to the maximum. Intuitive mechanics, convenient controls and addictive gameplay make this game a must-play.

Required controllers

This game requires a VR headset and two motion controllers to work. By moving the helmet, the player can sit down to crawl under obstacles or dodge enemy bullets, and various types of controller movements affect the height of the jump and the speed of the run. Using the buttons on the controllers, the player can cling to various objects in the game scene, as well as interact with objects and weapons.

Recommendation

The use of motion tracking sensors for the legs would help to transfer excess functionality from the hands and reduce the number of cases of motion sickness.



Figure 3.4: Stride game preview[16]

3.5 VR Chat

Main idea

The most popular social multiplayer online game for live communication for virtual reality devices. Thanks to a large community, convenient tools and opportunities for integrating your content, almost everything becomes possible in this game. By creating their own virtual worlds, players meet there to spend time together playing various games. Since the main purpose of this application is live communication, players use virtual avatars to visualize themselves in the virtual world. In order for these avatars to behave as realistically as possible, corresponding to the movements of the user in the real world, VR Chat supports most of the existing motion tracking systems.

Required controllers

The basic configuration for the game is a helmet and controllers, however, in addition to this, you can connect additional tracking devices, and their supported number is constantly increasing. For maximum immersion, players use motion capture suits for full body tracking. It is not uncommon to use AI face recognition.

Problems

Thanks to SteamVR and its extensive list of supported devices, the player can easily use their VR controllers. However, due to the lack of a common input standard, it is necessary to carefully configure the input system for each controller.



Figure 3.5: VR Chat game preview[17]

Game design concept

4.1 Goal

The main goal of this project is to write a game for virtual reality using different types of motion tracking sensors. To create a prototype, the game engine Unreal Engine 4 will be used, which already has the necessary plugins and a template for the development of VR-applications.

4.1.1 Criteria for VR app

Today's VR gaming platform is not as familiar to users as the mobile or PC realm. That is why when developing an application, special attention should be paid to the aspects that directly affect the user experience[26]. Usability is the starting point for any VR app to fully unlock the benefits of the platform. There are 2 main conditions that need to be met to create a good VR app:

- **High level of immersion** is the main advantage of VR. The more senses the player engages in his interaction with the virtual world, and the more believable these sensations are, the higher the degree of immersion. On the player's side this can be achieved through a large number of interactive mechanics. On the virtual world side, immersiveness is achieved through visual, auditory, and tactile sensations.
- **Simple and comfortable to use.** No matter how immersive the technical components of a VR experience are, a person's immersion will be ruined if they feel discomfort. One of the main obstacles for the user is the rejection of virtual reality by his nervous system because of the discrepancy between the real sensations and the information received through the controllers. One example of this behavior is motion sickness when traveling in virtual reality.

4.2 Main Idea

Before moving on to the creation of the game concept, I had to decide on the basic things on the basis of which the future prototype will be created. My goal was to create dynamic gameplay in virtual reality with hand and foot tracking support.

4.2.1 Inspiration - Subway

In order not to rack my brain for a long time to come up with fascinating game mechanics, I decided to analyze popular mobile games of the last decades. The main criteria for my search were:

- first or third-person camera view,
- dynamic gameplay with simple mechanics,
- easy to learn, hard to master concept.

My choice fell on a well-known game **Subway Surfers**, which fully met all the criteria of search. This game is an endless runner in the form of a lane with obstacles (trains and parts of the environment), from which the player must dodge. Controlling your character is very simple and is done with swipes on the phone. There are total 3 basic movements in the game - jump, rolls and side jump to change lanes.



Figure 4.1: Subway Surfers game preview[18]

4.3 Game concept

This concept is a classic example of endless runners, where the player's main objective is to achieve the highest score within a single run. The map is built procedurally from ready-made chunks and consists of 3 tracks for running with obstacles.

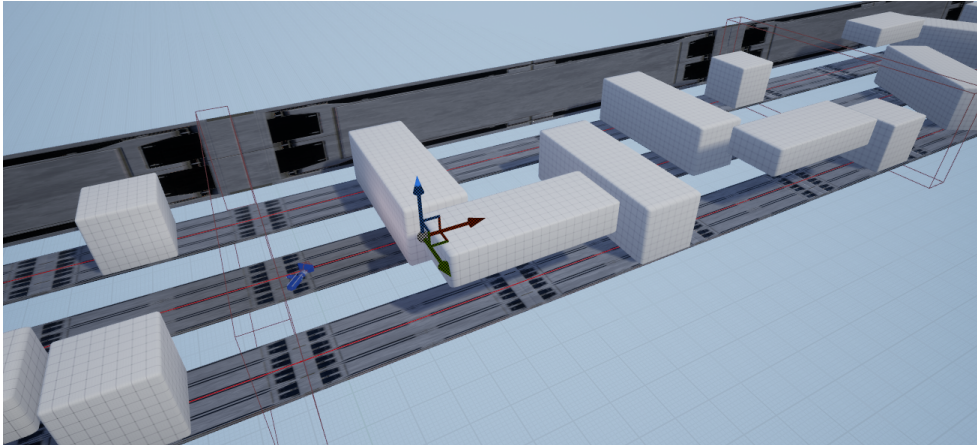


Figure 4.2: Example of a generated map

As the player runs through the map automatically and has no opportunity to stop, he must dodge obstacles to successfully progress through the map. Upon reaching the end of each chunk, the player receives a score showing the total number of chunks overcome. Also at this point the map is changed by spawning a new chunk and removing the last one.

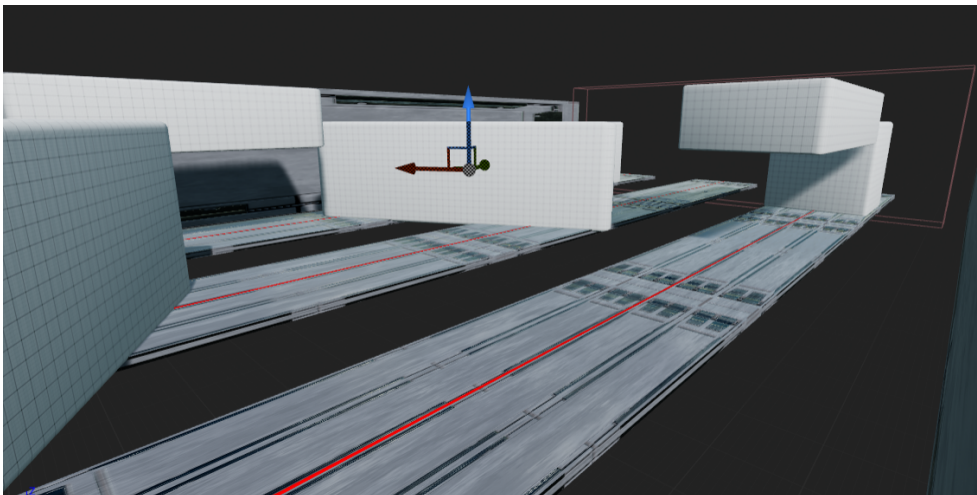


Figure 4.3: Example of a chunk with obstacles

4. GAME DESIGN CONCEPT

For dodging, the player has a number of special mechanics such as jumping, avoiding and accelerating. At each intersection with such an obstacle, the game reacts haptic effect. If the player fails to quickly dodge even after the controllers vibrate, he loses the game.

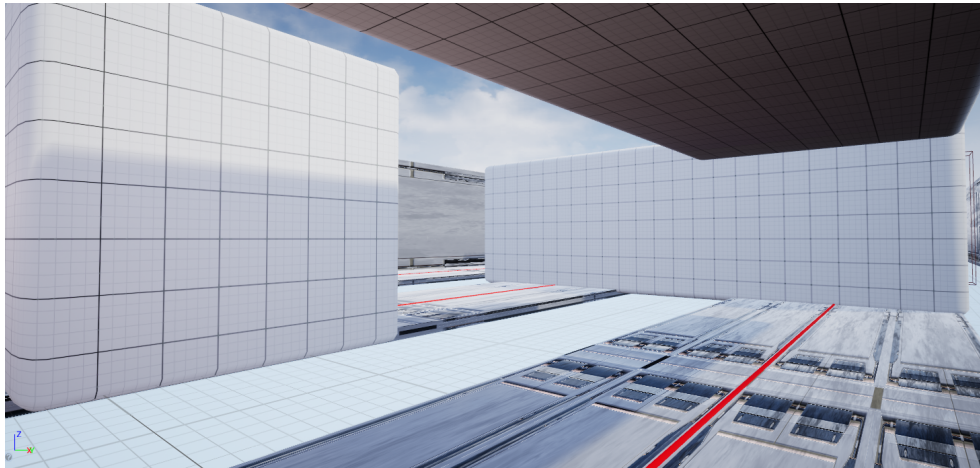


Figure 4.4: Screenshot of gameplay during the crouch

Moreover, the greater the distance the player travels, the smaller the distance between the obstacles and the higher his running speed. This complicates the gameplay, forcing players to break more and more records.

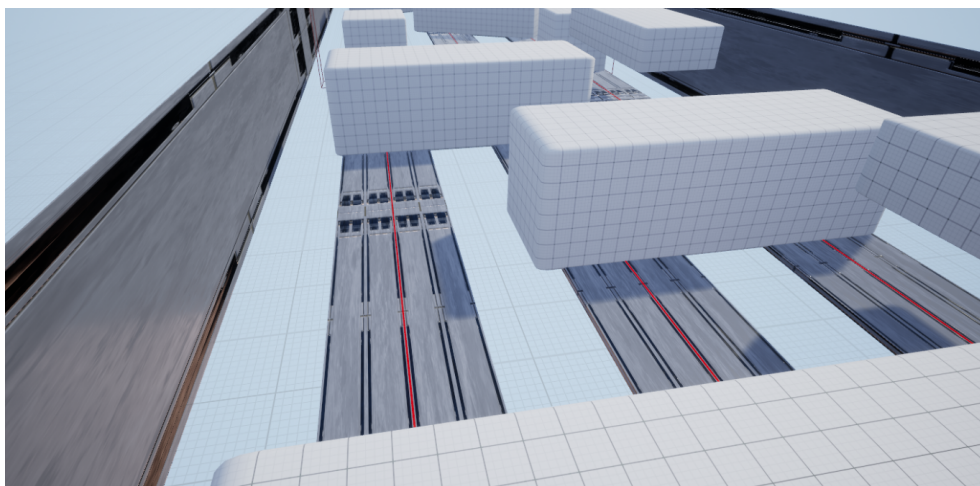


Figure 4.5: Screenshot of gameplay during the jump

4.4 Requirements

4.4.1 Functional

Functional requirements are, for the most part, responsible for gameplay and the range of possibilities for the player in the virtual world. Mechanics play a large part in making the gameplay fun. Below is a list of mechanics I have chosen, based on analysis and inspiration from existing projects.

Game mechanics

- Non-stop running — This is the main base of the game on which the other mechanics will be overlaid. The player, by constantly being on the move and changing positions on the map, will have to quickly assess and react to approaching obstacles.
- Endless level generation — In order to create a more interesting gameplay experience with support for the player's interest in replaying levels, I decided to make a procedurally generated map from pre-prepared chunks. This not only allows you to create countless possible routes to complete, but also to be sure of the quality of each individual section of the map.
- Increasing game complexity — The difficulty of the map and the speed at which a character moves through it will dynamically change over the course of the game, increasing the difficulty of the game. This will create not only a competitive effect, but also interesting gameplay for players with different skill levels.

Interaction mechanics

- Changing routs — One way of avoiding an obstacle in your path is to hop onto one of the three existing routes.
- Jump — Another option for a quick change of position but within the same rout is the jump. With it, you will be able to avoid some low objects in your path.
- Avoiding obstacles — This mechanic will allow the player to dodge and slip through to avoid obstacles that do not take up the entire width of the track.
- Acceleration — You can use the acceleration to go through certain sections of the route much faster, break through certain objects and jump over taller objects.

4.4.2 Non-Functional

Non-functional requirements will describe the correct operation of a VR game in terms of usability. The main task was only correct realisation of functional requirements for virtual reality. My goal was to make the controls as simple and intuitive as possible, to achieve a sufficient level of immersiveness. To achieve it, I decided to abandon the buttons on the handheld controllers as analog input sources, and implement all the transferring to specific sensor movements. Since Subway Surfers is a dynamic endless runner where the player's success depends directly on the speed of his reactions, the control had to fulfill the following requirements:

- Comfortable — One of the most important control requirements for virtual reality games. As the main source of movement, it influences the perception of information coming from our eyes, ears and body. The greater the synchronisation between real-world actions and the image in the glasses, the less it is rejected by the player's body. Comfortable handling will not cause the player to become motion sick.
- Intuitive — The more intuitive the controls are for the player, the lower their threshold of entry into the game and the less time it will take to learn. The intuitive controls allow you to react properly to emergencies on a reflexive level, which my game will undoubtedly need.
- Easy to activate — This rule is particularly important for virtual reality games, where the player is exhausted much faster. Since my game is designed for long and repetitive play sessions, in order for the player not to tire physically too quickly, the controls must be easy to use.
- Easy to combine — The variety in the construction and completion of the game levels may force the player to perform several actions at once (e.g. changing the line while jumping). This is why all game inputs must be combinable and mutually reinforcing.
- Accuracy in recognition — Since this is primarily a VR game with non-standard inputs and a dynamic game environment with increasing levels of difficulty, the controls must recognise the player's movements as accurately as possible to avoid accidental inputs.
- Have a fast response — The higher the difficulty level, the greater the number of obstacles and the higher the speed of the player. This forces the player to make quick decisions to avoid a collision. The controls must be as responsive as possible in such gameplay conditions.

4.5 Chosen controllers and input system

Based on all the requirements and my analysis of motion tracking systems, I selected the following types of controllers to implement my prototype.

4.5.1 VR helmet — Oculus Quest 2.

Game functions

In my prototype game, the helmet was not only to act as a stereoscopic display, but also to enhance the player's movement capabilities. For example, tracking the position of the player's helmet would allow us to implement the mechanics of dodging obstacles within a single track.

Reason of choose

As I needed to monitor all 6 degrees of freedom, the choice was between a portable helmet and a stationary helmet. Due to cost and not wanting to be tied to a specific gaming space, I opted for the Oculu Quest 2 portable helmet. I would also like to port the prototype to a mobile helmet platform in the future, making the apps completely portable.

Input type

The main difficulty of the game is the obstacles, on collision with which the player loses. Collisions are calculated between the collisions of the obstacles and the collision of the player. The player's collision is a capsule with a small radius and an updatable height equal to the distance between the helmet and the floor. The capsule is on the track and its position also dynamically changes following the movement of the helmet.

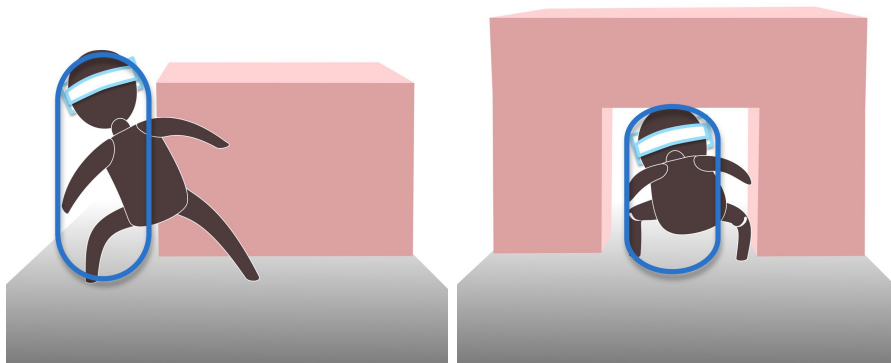


Figure 4.6: Demonstration of how character collision works

4.5.2 VR motion controllers — Oculus Touch

Game functions

The controllers are used as the main control for making fast and long jumps. Whether jumping from track to track or jumping on one of them, this kind of locomotion is based on short and fast swings of the controllers, as if the player is pushing away from something.

Reason of choose

As with the helmet, all 6 degrees of freedom are required to determine the position of the controllers in space for further motion recognition. The Oculus Touch motion controllers supplied with the helmet were fully suitable for this purpose for me. Using the built-in cameras, the helmet itself detects the position of the controllers in space and calibrates the values from the sensors.

Input type

Since controllers take over 2 movement mechanics at once, the movements recognised as input for the game must be well matched to the non-functional requirements. I decided to define these types of movements using the controllers' vector speed. By tracking its magnitude along the different world axes, I can not only clearly distinguish between different types of input, but also combine them with ease. For example, for an upward jump, information about the vector speed of the controllers on the world up axis is collected and added together. As soon as this value exceeds a certain limit, a jump will be made.

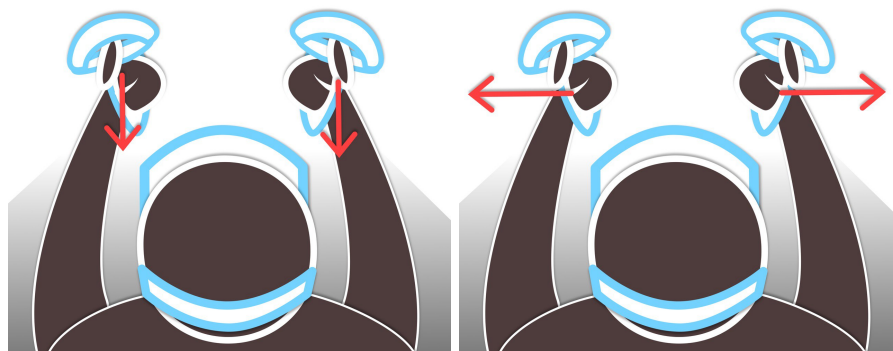


Figure 4.7: Demonstration of the principle of jump activation

4.5.3 Additional sensors for tracking legs — Kat loco S

Game functions

The last part of the interaction mechanics, namely acceleration, should be under the control of the foot controllers. The foot movements will make it easier to control the hand controllers, but also make the movements more natural and comfortable. To test the effectiveness of this method, I also implemented an alternative method using the hand controllers for comparison.

Reason of choose

Cyber Shoes could have been the ideal solution for me due to their absolute portability in direct connection and transferring input to the Oculus Quest 2 headset. But they could only be used in a seated position, which broke the other mechanics by limiting the player's movement. Since I don't need all 6 degrees of freedom for on-the-spot running recognition, I opted for the cheaper Kat Loco S controllers. Needing to directly connect the base station of these controllers to the computer, I unfortunately lost the portability of the solution. Instead of the mobile version of the application, I was temporarily forced to make a desktop version due to a lack of the necessary technology. However, a big plus for me was the official plug-in for Unreal Engine 4, which facilitates working with these controllers.

Input type

This type of input is the easiest both to implement and to calculate. The acceleration is calculated from the magnitude value of the angular velocity vector. In order to accelerate, the player has to start running in place, thereby rotating the controllers attached to the legs.

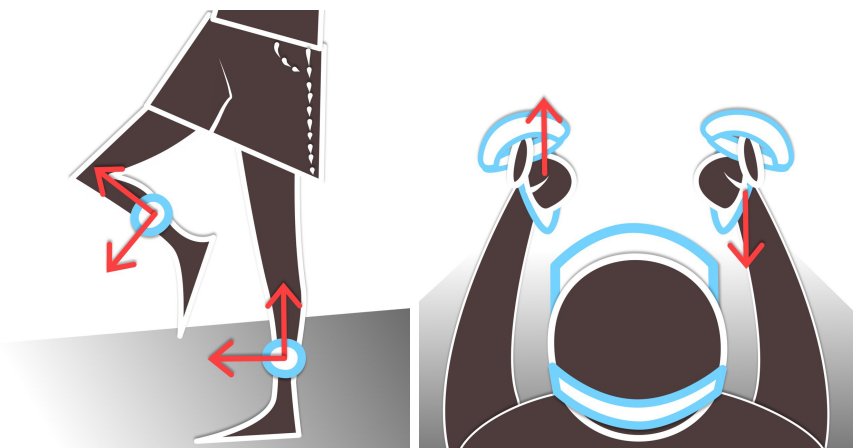


Figure 4.8: Demonstration of the principle of acceleration activation

4.6 Sequence diagram

Sequence diagram describing the process of interaction of the main game components within the game cycle.

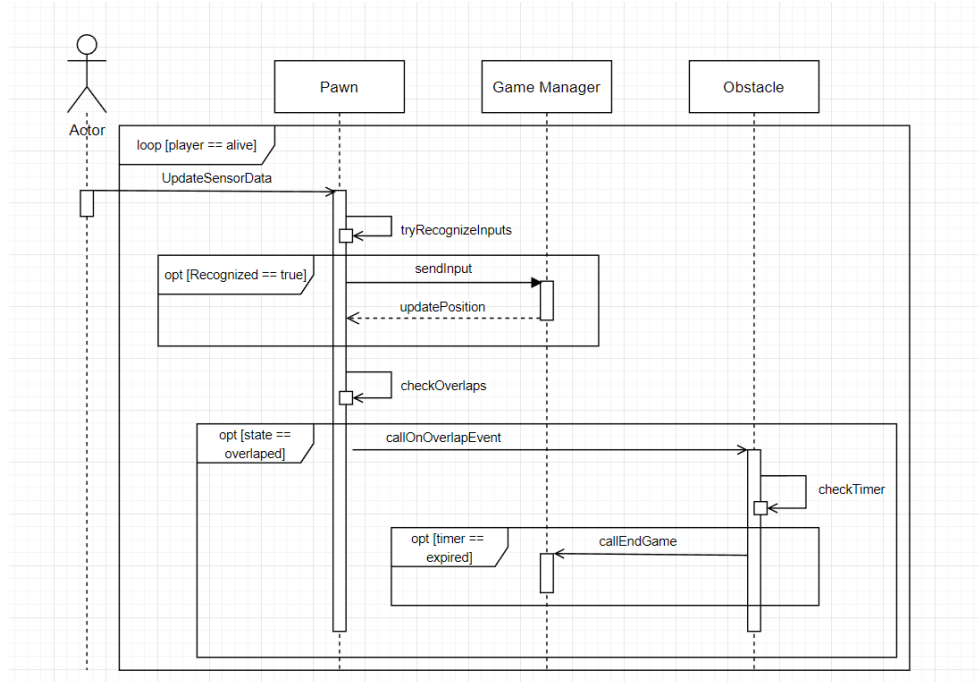


Figure 4.9: Sequence diagram

Implementation

In this chapter I will briefly describe how I implemented the basic functional requirements of the prototype using the Unreal Engine 4 and its scripting system called Blueprints.

5.1 Plugins

To work with the basic components of virtual reality, like hand controllers and a helmet, I need to enable 2 plugins:

- OpenXR — as a standard for working with VR controllers and the events received from them.
- Oculus OpenXR — to connect additional functionality and extended work with the Oculus headset.

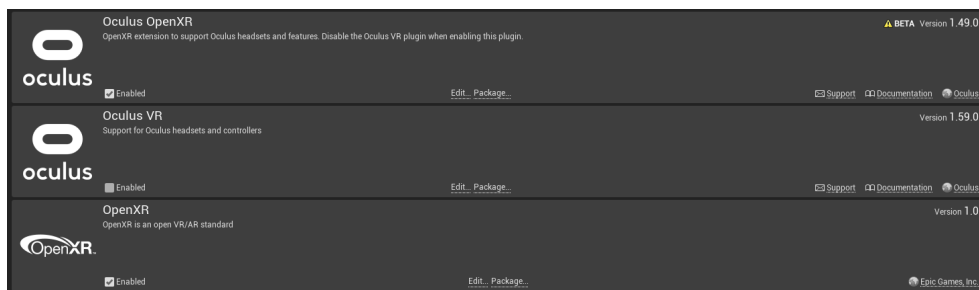


Figure 5.1: List of plugins used to work with VR controllers

5.2 Base world mechanics

5.2.1 Auto run and increase game difficulty

The Game Manager calculates the game map speed based on the player's current base speed and the difficulty factor. The increase in difficulty depends directly on the elapsed time since the start of the game session.

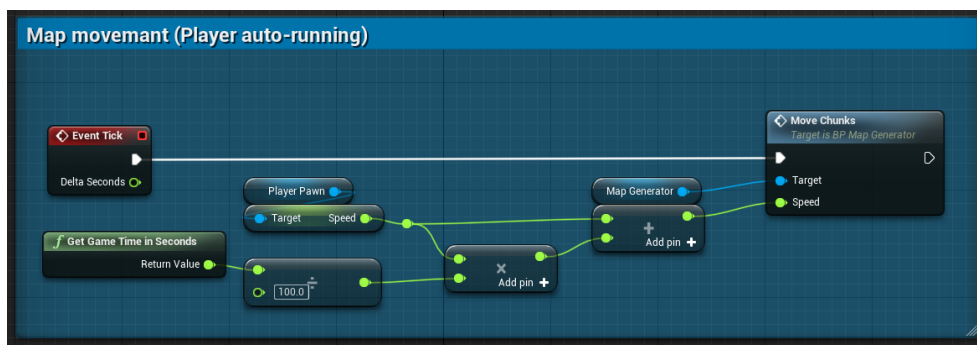


Figure 5.2: The logic of auto-running and increasing the game's difficulty.

5.2.2 Procedural generation

Procedural generation updates the state of the game level in real-time. Each time the player reaches the end of a chunk, the game manager sends an event to update the level. This includes deleting passed chunks, creating new ones and updating all references. When creating a new chunk, the game manager randomly chooses one of the Master Chunk's successor classes and docks it to the last chunk. An array of chunk classes is defined in the game manager before the game starts.

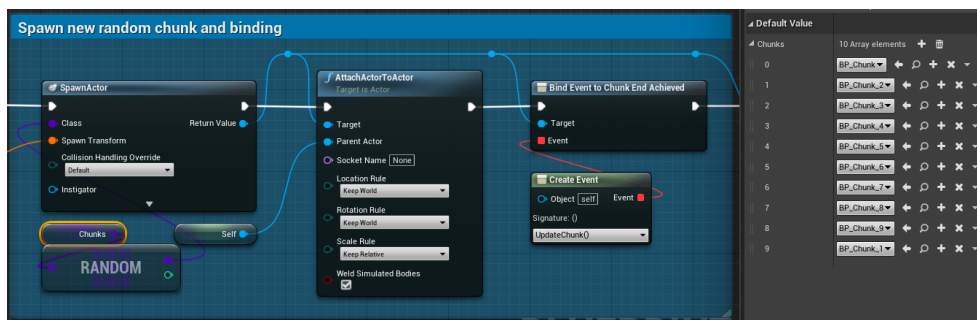


Figure 5.3: Procedural map generation from ready-made chunks.

5.3 Controllers input tracking

Each game frame updates the status of all controllers, including their position. Based on the data received, further calculations are made to determine the necessary values on the basis of which tracking is performed and the type of input is determined. The entire tracking process is a primitive state machine, where each type of controller is handled separately and independently of the others. It also takes into account not only their position, but also the status of the tracking system as a whole in order to correct the performance and avoid false inputs.

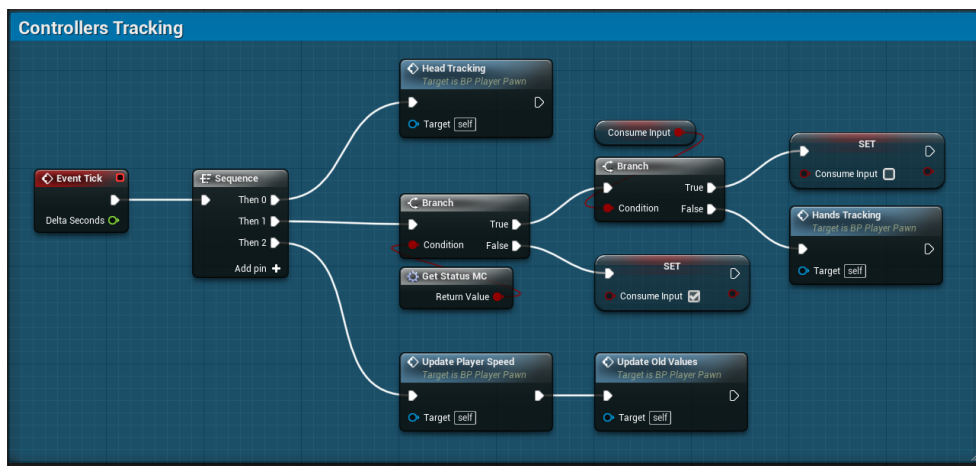


Figure 5.4: State machine for logic tracking controllers.

In order to work with KatLocoS controllers, an official plug-in must be installed and enabled, allowing access to data received from the controllers by the base station. Other Oculus controllers are supported by OpenXR by default.

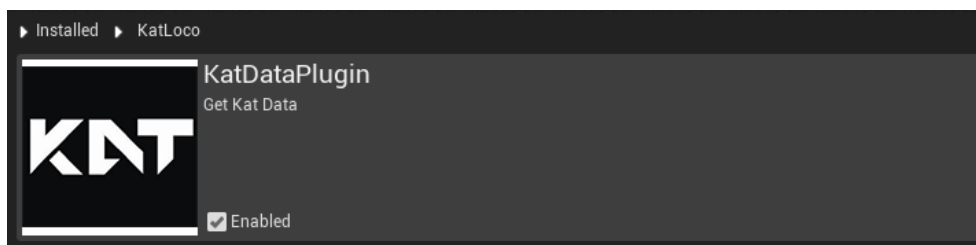


Figure 5.5: Plugin for collecting data from foot controllers

5.3.1 Acceleration

One of the main results of the leg tracking process, is to calculate the speed of the controllers. Based on this value, the player's speed is updated. Running intensity data is obtained by calling the *KATLocoWalkSpeed* function. Eventually the player's current speed is used by the game manager to update the game level position.

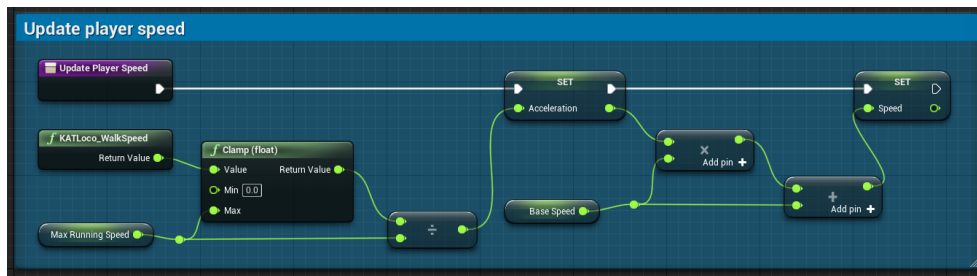


Figure 5.6: Acceleration calculations based on data from the foot controllers

5.3.2 Dodging

The entire process of calculating and updating the player's collision is based on the virtual reality glasses transformation data. The movement of the helmet along the XY axes thus directly determines the position of the capsule in the game. And data about the height of the helmet above the floor is used to calculate the height of this capsule. In this way it is possible to determine the position of the player's body for further collision handling.

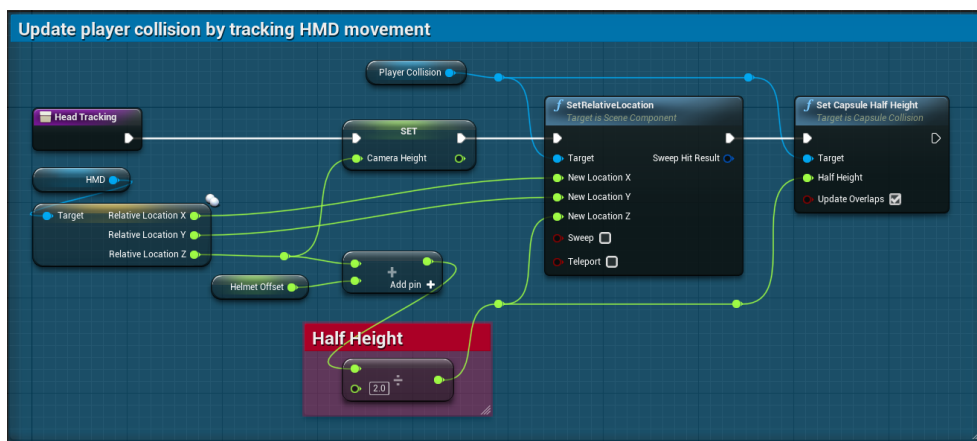


Figure 5.7: Implementation of a dynamic player collision

5.3.3 Jumps

All types of hand input are recognised by calculating the velocity of each controller. By comparing these values and projecting them onto specific axes, it is possible to determine the type of user input.

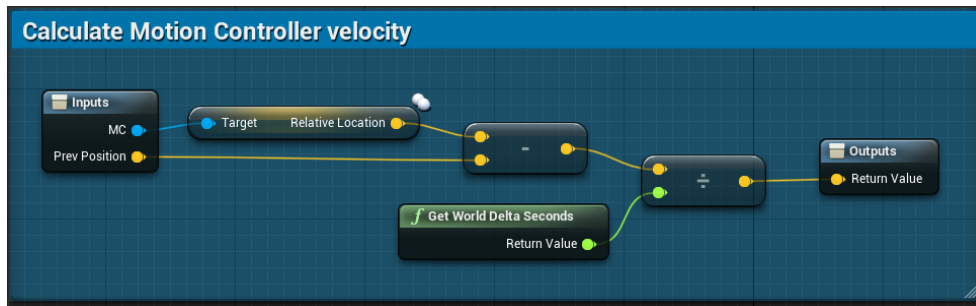


Figure 5.8: Calculating the velocity of the hand motion controllers

Jump events are triggered when the minimum required velocity in one of the specific axes of both controllers is reached. The required minimum values are stored in the *RequiredJumpSpeed* and *RequiredStraifSpeed* variables.

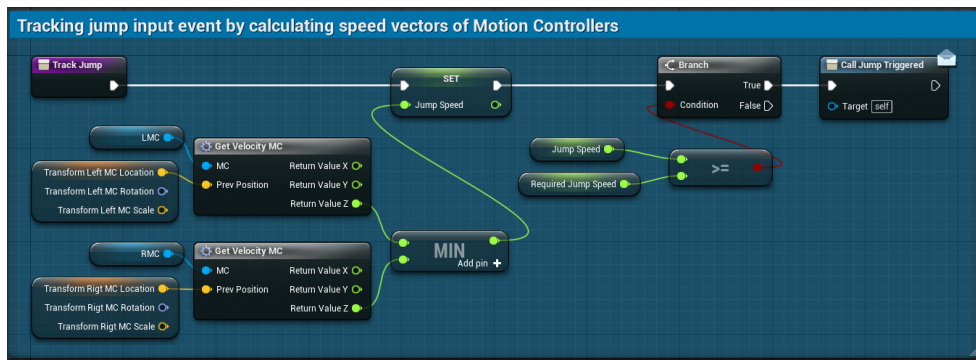


Figure 5.9: Implementation of upward jump tracking.

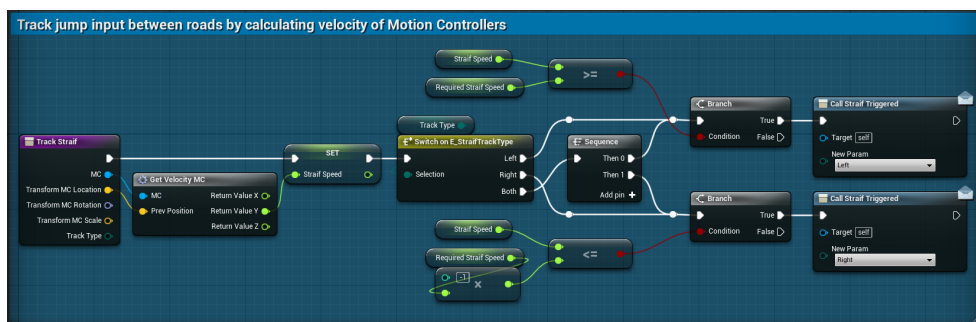


Figure 5.10: Implementation of sideways jump tracking.

5. IMPLEMENTATION

5.3.4 Hits reaction

Each of the obstacles individually handles collisions with a player. When a player's collision capsule crosses an obstacle collision, a timer is started. When the timer expires, the end game event is triggered and the player loses the game. However, if the player manages to exit the obstacle collision within the allotted time, the timer is turned off. The hand controllers also trigger a haptic effect when they encounter an obstacle.

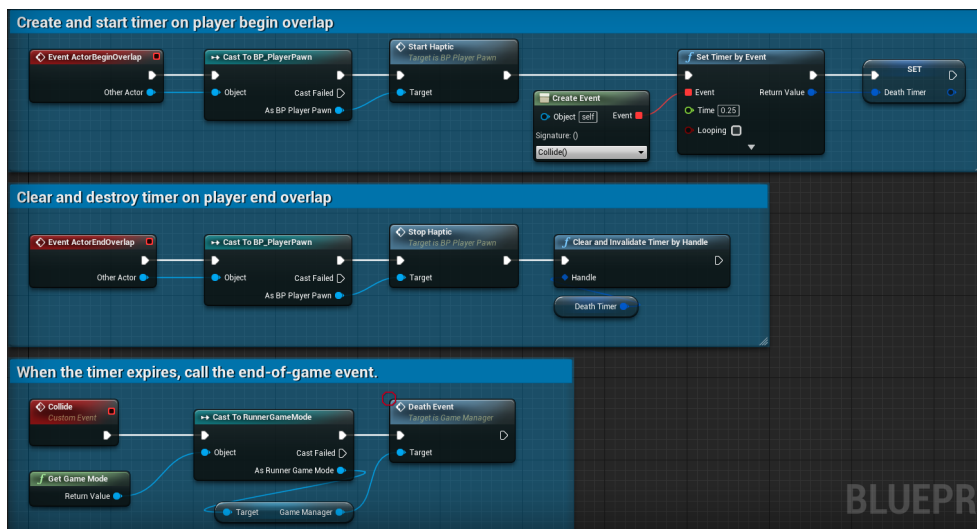


Figure 5.11: Player collision handling

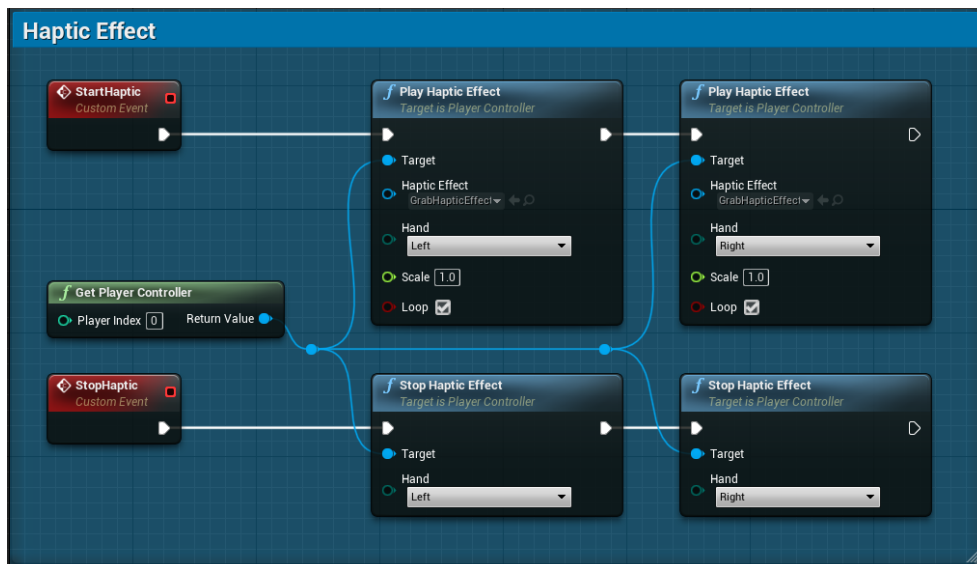


Figure 5.12: Tactile effect implementation

Testing

Since the game is in the prototyping phase, I did not touch the graphics or the user interface. My main goal for testing was to determine the functionality and usability of the controls and also to compare two types of locomotion using different combinations of motion-tracking sensors.

6.1 Mechanic testing results

Below will be the result of the prototype test in the form of a table affecting all the main interactive mechanics. The column headings in the form of numbers indicate the corresponding non-functional requirement. A list of numbers and their corresponding requirements is given below:

1. comfortable,
2. intuitive,
3. easy to activate,
4. easy to combine,
5. accuracy in recognition,
6. have a fast response.

The assessment was carried out among 14 people who tested, with or without previous experience of virtual reality. The rating scale is given below:

- A. Perfectly fulfils the requirement.
- B. Has minor disadvantages.
- C. Has significant disadvantages.
- D. Does not fulfil the requirements.

Mechanic testing							
Controllers for acceleration mechanic:	1.	2.	3.	4.	5.	6.	Total
Jump							
Oculus Touch	B	A	A	B	B	A	B
Kat Loco S	C	A	A	A	A	A	C+
Side jump							
Oculus Touch	A	A	A	B	B	A	B+
Kat Loco S	A	A	A	A	A	A	A
Avoiding							
Oculus Touch	A	A	A	A	A	A	A
Kat Loco S	A	A	B	B	A	A	B+
Acceleration							
Oculus Touch	B	A	A	B	B	A	B
Kat Loco S	A	A	B	B	A	A	B+
Overall							
Oculus Touch	B	A	A	B-	B-	A	B
Kat Loco S	C+	A	B	B	A	A	B

Table 6.1: Overall testing results

As you can see from the results, both types of locomotion performed quite well, in the very dynamic gameplay of the prototype. In short, the following summary can be made about the need for additional leg tracking sensors:

- **Without the use of additional controllers**

Using hand controllers for acceleration proved to be more comfortable and easier due to greater player stability and less physical stress. However, due to the large number of mechanics involving these controllers, it became more difficult to combine movements, and the false input rate increased.

- **With the use of additional controllers**

In addition to the easier combination of inputs and their more accurate recognition, the immersion effect has increased. Testing this method was complicated by the presence of a cable connected to the helmet. This severely limited the space for the player's movements. The most serious problem for users was the discomfort of activating acceleration during a jump. Jumping is one of the hardest types of locomotion to perceive in virtual reality. While in the air, the player needs to have a stable footing, but when running in place, balance is lost. For some players, for the first time, this can cause motion sickness.

6.2 Gameplay testing results

During the design phase, I made many decisions that affected not only the in-game functionality, but also the non-functional requirements. After testing was completed, I also received a lot of feedback from users on the success of certain decisions. These are the main questions for me that I got answers to:

- **How much were you involved in the game?**

Almost all of the testers described the gameplay as quite immersive and engaging. The gameplay mechanics were quite dynamic and had, in their opinion, potential for further development.

- **What do you think about the control without pressing any buttons?**

This kind of control also affected the immersion effect for the better. Giving up the buttons allowed players to use more of their body and respond better to some game moments.

- **What do you think about the game concept in general?**

The concept may not be new, but it has no analogues in virtual reality. I received a lot of ideas and suggestions for improving and diversifying the game mechanics, as well as the visual component of the game.

Conclusion

In conclusion, I would like to summarize the work done in my bachelor's project:

1. I analyzed plugins and frameworks for creating a VR application on the UE4 engine.
2. Next, I described the types of locomotion, as well as the types and principles of operation of motion tracking systems for its implementation in virtual reality.
3. I conducted an analysis of existing motion controllers for virtual reality and their scope.
4. Based on the information received, I analyzed existing VR applications for the use of different motion tracking systems and the possibilities for their improvement.
5. Based on all the analysis carried out, I compiled the first concept of the game, as well as functional and non-functional requirements for it.
6. Following the requirements, the controllers and the input system were defined, as well as the main architecture of the application.
7. Next, two prototypes were created using different motion tracking systems for further comparison.
8. And in the final, a thorough testing of the prototype was carried out for the feasibility of using the selected types of motion tracking systems and the game concept as a whole.
9. The result of the work was 2 fully working prototypes using different options for tracking systems.

CONCLUSION

Based on the results of testing, I concluded that the use of leg tracking systems has its advantages in virtual reality. Their full implementation into the standard of the tracking system for virtual reality is only a matter of time and technology development. The trackers I used had a number of limitations and inconveniences, which also affected the final result. With the use of more advanced systems, as well as the results of this work, a better prototype can be achieved.

Bibliography

- [1] Heaney, D. *How VR Positional Tracking Systems Work - UploadVR [online]*. UploadVR, 05 2019, [cit. 2022-04-24]. Available from: <https://uploadvr.com/how-vr-tracking-works/>
- [2] Ago, L. *[IT STORY] AR Glasses 'MIX' [online]*. Steemit, 06 2018, [cit. 2022-04-24]. Available from: <https://steemit.com/busy/@leomichael/it-story-ar-glasses-mix>
- [3] developer.mozilla.org. *WebVR concepts - Web APIs — MDN [online]*. 06 2022, [cit. 2022-04-24]. Available from: https://developer.mozilla.org/en-US/docs/Web/API/WebVR_API/Concepts
- [4] Chow, B. *Virtual reality wins Christmas [online]*. AllThingsVR, 12 2016, [cit. 2022-04-24]. Available from: <https://medium.com/allthingsvr/virtual-reality-wins-christmas-e6fc3a001ebd>
- [5] Melnick, K. *HTC Vive Pro Secure Offers Security-Focused VR For Government Agencies [online]*. VRScout, 10 2020, [cit. 2022-04-24]. Available from: <https://vrscout.com/news/htc-vive-pro-secure-announced/>
- [6] Olšan, J. *Autonomní VR brýle Oculus Quest 2: vydání, specifikace, cena [online]*. Cnews.cz, 09 2020, [cit. 2022-04-24]. Available from: <https://www.cnews.cz/facebook-vr-bryle-oculus-quest2-uvedeni-specifikace-cena-datun>
- [7] to VR, R. *Road to VR [online]*. Road to VR, 02 2019, [cit. 2022-04-24]. Available from: <https://www.roadtovr.com/>
- [8] Skarredghost. *HaptX Gloves hands-on: one of a kind haptic device for VR [online]*. The Ghost Howls, 11 2021, [cit. 2022-04-24]. Available from: <https://skarredghost.com/2021/11/18/haptx-gloves-hands-on-review-2/>

BIBLIOGRAPHY

- [9] Ultraleap. *Digital worlds that feel human — Ultraleap [online]*. www.ultraleap.com, [cit. 2022-04-24]. Available from: <https://www.ultraleap.com/>
- [10] KatVR. *KAT VR Official Site - Virtual Reality Locomotion Technologies [online]*. KATVR, [cit. 2022-04-24]. Available from: <https://www.katvr.com/>
- [11] Lang, B. *HTC Announces Face-tracker for Vive Pro and Vive Tracker 3.0 [online]*. Road to VR, 03 2021, [cit. 2022-04-24]. Available from: <https://www.roadtovr.com/htc-vive-facial-tracker-3-0-announcement-release-date-price/>
- [12] Record, C. G. *OptiTrack Motive 3.0 is now available [online]*. CG Record, 07 2021, [cit. 2022-04-24]. Available from: <https://cgrecord.org/optitrack-motive-3-0-is-now-available/>
- [13] VR, T. *Toast VR — Richie’s Plank Experience [online]*. toast.games, [cit. 2022-04-24]. Available from: <https://toast.games/>
- [14] SUPERHOT. *SUPERHOT - Time moves only when you move [online]*. 2018, [cit. 2022-04-24]. Available from: <https://superhotgame.com/>
- [15] Wolfe, H. *New arena Lockport Bridge comes to The Unspoken today [online]*. GAMING TREND, 01 2017, [cit. 2022-04-24]. Available from: <https://gamingtrend.com/news/new-arena-lockport-bridge-comes-to-the-unspoken-today/>
- [16] games, J. *STRIDE developed by Joy Way [online]*. stride.joyway.games, [cit. 2022-04-24]. Available from: <https://stride.joyway.games/>
- [17] Hamilton, I. *VRChat Raises 80 Million To Prepare For Social VR Growth [online]*. uploadvr.com, 06 2021, [cit. 2022-04-24]. Available from: <https://uploadvr.com/vrchat-raises-80-million/>
- [18] Games, K. *Subway Surfers (Online Game) Unblocked — Kiloo.com [online]*. www.kiloo.com, [cit. 2022-04-24]. Available from: <https://www.kiloo.com/subway-surfers/>
- [19] Epic Games. *OpenXR [online]*. [cit. 2022-04-24]. Available from: <https://docs.unrealengine.com/4.26/en-US/SharingAndReleasing/XRDevelopment/OpenXR/>
- [20] Epic Games. *Unreal Engine 4.27 Release Notes [online]*. [cit. 2022-04-24]. Available from: https://docs.unrealengine.com/4.27/en-US/WhatsNew/Builds/ReleaseNotes/4_27/

- [21] Epic Games. *Input with OpenXR [online]*. [cit. 2022-04-24]. Available from: <https://docs.unrealengine.com/4.26/en-US/SharingAndReleasing/XRDevelopment/OpenXR/Input/>
- [22] developer.oculus.com. *VR Locomotion Design Guide — Oculus Developers [online]*. [cit. 2022-04-24]. Available from: <https://developer.oculus.com/resources/bp-locomotion/>
- [23] Barnard, D. *Degrees of Freedom (DoF) [online]*. Virtualspeech.com, 09 2018, [cit. 2022-04-24]. Available from: <https://virtualspeech.com/blog/degrees-of-freedom-vr>
- [24] Council, N. R. *Virtual Reality: Scientific and Technological Challenges*. Washington, DC: The National Academies Press, 1995, ISBN 978-0-309-05135-4, doi:10.17226/4761, [cit. 2022-04-24]. Available from: <https://nap.nationalacademies.org/catalog/4761/virtual-reality-scientific-and-technological-challenges>
- [25] VR, R. t. *Overview of Positional Tracking Technologies for Virtual Reality [online]*. Road to VR, 06 2014, [cit. 2022-04-24]. Available from: <https://www.roadtovr.com/overview-of-positional-tracking-technologies-virtual-reality/>
- [26] Dealessandri, M. The best practices and design principles of VR development [online]. 04 2020, [cit. 2022-04-24]. Available from: <https://www.gamesindustry.biz/articles/2020-04-01-the-best-practices-and-design-principles-of-vr-development>

Acronyms

VR Virtual reality

AR Augmented reality

UE4 Unreal engine 4

DoF Degrees of freedom

Contents of enclosed CD

	readme.txt	the file with CD contents description
	exe	the directory with executables
	android.....	android package for oculus quest 2
	windows	windows x64 package
	src	the directory of source codes
	prototype.....	implementation sources
	thesis.....	the directory of \LaTeX source codes of the thesis
	text	the thesis text directory
	thesis.pdf.....	the thesis text in PDF format
	gameplay.mp4	gameplay recording