Peter Virtanen

ACCESSIBILITY SIMULATOR IN VR-ENVIRONMENT

Master’s degree in Welfare Technology

2019

samk
Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

ACCESSIBILITY SIMULATOR IN VR-ENVIRONMENT

Virtanen, Peter
Satakunnan ammattikorkeakoulu, Satakunta University of Applied Sciences
Degree Programme in Welfare Technology
September 2019
Number of pages: 55
Appendices: 0

Keywords: virtual reality, augmented reality, Unity, Unreal, serious games

---

Accessibility requirements are often met on paper but in practice, some problems may still occur. The purpose of this thesis was to create a simulator, that allows testing different environments accessibility issues in virtual reality. The simulator is intended to work as a testing tool that can help to test buildings and its accessibility before the building is constructed. This thesis goes through subjects like human resources, hardware and technical requirements, that are needed to create this simulator.

Testing is done in a virtual reality environment with a wheelchair. Virtual wheelchair reacts to the environment like a real wheelchair. Environments that are used for this simulator must be placed precisely. Every threshold, door and doorknob must be in right place to make test results of this simulator valid. The thesis also discusses the problems and demands that are faced when modelling extensive environments like shopping centers or school buildings.

Virtual reality can be very immersive, and this thesis undergoes what factors make the virtual reality experience immersive. The simulator is required to be immersive to make the test experience memorable. A memorable experience is much more personal than a simple test report.

This accessibility simulator was a big enthusiastic project that gathered a lot of information about how to create such a simulator. With the background work done, it is now easy to continue the development towards the essential production stage. Demos that were made for this project can work as a proof of the concept but should not be used as a base for the production version of the accessibility simulator.

Next phase of the development is to make a working prototype and test it in a collaboration with accessibility professionals. Prototype of an accessibility simulator needs to be tested with a real-world counterpart of an environment like a shopping center. According to test results with the prototype, it may be developed further towards the first public release.

SISÄLLYS

1 INTRODUCTION

1.1 Guaranteeing accessibility of every building

Technology has evolved fast in last twenty years. Today we have lot of technological innovations in use. Often, we do not even realize how well we have adapted technology in our lives and how much technology eases out lives. Smartphones, computers, the internet, wearable technology and many more innovations are all making our lives easier. We often take these granted and that is why modern people does not pay a lot of attention to these technological marvels.

Virtual Reality, VR for short is one technological marvel. According to Statista virtual reality and augmented reality had a 18,9 billion U.S. dollar market globally on 2019. Forecast for 2023 is that global market for VR increases to 160 billion U.S. dollars (Statista, 2019). If the forecast is to believe VR and AR are here to stay. Among consumers VR is little controversial. Almost everyone sees the possibilities of VR and enjoy the technological demos that has been shown to them but often see that VR is unreachable technology because of its high price tag. Experiencing a high-

quality VR-game by major publisher at home means a big investment for powerful PC and VR kit like HTC Vive or Oculus Rift (Thinknum Alternative Data, 2018).

Although VR is costly it does not mean that it is unreachable technology for companies. Companies usually have a bigger investment budgets towards technology than an average consumer does. This thesis is a research to create a testing tool for accessibility testing. Main focus group for this kind of product are companies that design and construct building and public areas. In following chapters need for this kind of testing tool or product is explained better. In short accessibility is a requirement not just nice to have feature. This is why a next generations accessibility testing tool for professionals is so important to be researched. This thesis tells about planning, analyzing requirements, programming and human resources that are needed for this kind of system. This simulation is not aimed for consumer markets, but it has also its separate gamified simulation part that is purely designed for consumers. More about why this product should meet consumer markets in chapter two.

Simulation like this could change the accessibility testing of the future. A one more technology innovation that makes our lives better, but we do not even notice that we are using it or enjoying the outcome of it.

1.2 Accessibility in Finland

It is clear that accessibility is on a different level than it was 80 years ago. Back then there was not many standards for accessibility and especially there was no technology like VR to test the accessibility issues with. Many older buildings were and still are inaccessible. These buildings don’t have elevators or ramps for disabled people. It certainly doesn’t mean that back then there were no disabled people. Second world war had just ended. This unfortunate era left many veterans and civilians disabled. At the end of the 1940s Sotainvalidien Veljesliitto, a union for war invalids had 50 000 members (Kadettikunta Ry, 2014). That means that a need for accessibility solutions is not a new thing, there has been a deed for a long time.

Though accessibility had a big need back then things didn't change too quickly. Especially in 1960 to 1980 built districts apartment buildings were built without including an elevator. At 2014 in those same apartment buildings lives over 100 000 over 65 years old people. In Finland at 2014 still every second apartment building didn't have an elevator (ARA, 2014).

At the 21th century things have started to turn better. At 2008 a model house for accessible workplace finished. It is called Esteetön toimitalo and it was built in Helsinki. At 2010s Invalidiliitto has a plan to have equal and accessible Finland. Among accessible development it also aims to change attitude and communication limitations (Invalidi Liitto RY 2019c).

1.3 Equality in work life

Whether a person has or hasn't a disability one has a right to be treated equally. This is backed with Non-discrimination Act that was entered into force on 1 January 2015.

In short, this act requires that employer needs to promote equality among employees and job applicants. Employer also has special requirements towards disabled people. Employer must treat disabled applicants equal to non-disabled applicants. Also, employer must make sure that the disabled employees can overcome their work tasks equally compared to non-disabled (Occupational Safety and Health authority in Finland, 2019).

1.4 A better place

This thesis and the simulation are trying to create a better world. It is shocking, that how little, common people know about the accessibility or pay attention to it. Idea is to generate two different products or serious games. One is for making accessibility easier to understand with a gamified fun to play game and other is a testing tool for professionals.

Construction of buildings is costly and construction companies do not create accessibility issues on purpose. It is understandable that although construction companies have good blueprints and guideline sometimes plan have flaws. As an example, little miscalculation by an architect or designer in design program or unstable soil may create huge threshold between two parts inside of a building. This could be tested with a simulation tool like described in this thesis. Even scenarios of unstable soil and consequence of that could all be seen on simulated environment. Caused accessibility issues could then be seen before constructing and also make a plan or plans how to fix this foreseen accessibility issue. In this example maybe a ramp would fix the issue if it is caused by unstable soil or if it is just a miscalculation it can be seen before constructing this error.

2 IDEATION

2.1 Why this project and planning was so important?

What comes to accessibility, today situation is visible in our daily life. In public buildings and business spaces path to front entrance needs to be accessible. Usually this means some kind of ramp that has to be hard, flat surface that is non-slipping even if it is wet (Invalidi Liitto RY 2019b).

The national building code of Finland got new requirements in 2005. There are many rules and requirements for accessibility. New building code wraps up movement requirements for car parks, joining building, elevators, ramps, accommodation, gathering spaces and hygiene premises (Finlex, 2004). New requirements of course force constructor to meet these requirements. Unfortunately, that does not always mean that all buildings would get as easy access with wheelchair as a people with no disabilities.

Image 1. Accessibility by regulations.

Image 1. exhibits an entrance ramp that fails to comply regulation or common sense. Ramp to this recreational space was at least more than 12 meters long. It most certainly fills requirement of 8% declination as requirement requires. Requirement also requires that this declination can continue only for 6 meters. This ramps declination to one way is more than 6 meters. By eye inspection image 1. ramp does not fill that requirement. Also, noticeable accessibility issue is that snow piles up on the start of the ramp and that makes it also slippery. One may ask why this kind of ramp was created in the first place. On paper all this may have looked good. Reality and the finalized building shows that ramp is poorly designed, and it does not totally fill its purpose of being accessibility aid as intended.

Ramps like shown in Image 1. inspired this thesis. Although there are many requirements, laws and rules for accessibility it is hard to test building before it is built. This thesis and its final product, accessibility simulator is a tool to test buildings before those are even built. Good ideas on paper may be noticed to be bad just using the simulator. There is also possibility to create more imperceptible accessibility solutions than before and test those in practice. Even Image 1. ramp could have been tested and seen as too long in practice. Maybe an elevator would have solved this better or some other accessibility solution.

2.2 Vision about the final product

This thesis is about generation of a product that is used to test accessibility issues. It is a simulator that helps to test environments like offices and shopping centers for any accessibility issues. This is a tool but not only for professional use. Simulation is also an eye opener for general public. This simulator has many gamified features which should give an easy approach for understanding accessibility in general. While “playing” the game, aim is to provide a player with better understanding of accessibility issues. This product has also more serious game part that is aimed purely to test environment accessibility issues without any gamified parts. This more serious part is aimed for professionals and more gamified part is for general public.

Gamified levels or environments contain simple tasks like “Go to bathroom”. These simple missions will very fast make clear if environment has any accessibility issues. As an easy example accessibility issue like closed door or very high threshold becomes noticeable for player, because it prevents player for using certain route. After an accessibility issue has been found it can be marked as an issue within the simulation or points can be gained from overcoming that obstacle.

2.3 Basic issues

Healthy regular person, that has no disabilities things in this world are almost always easier. Though, when normally functioning person notices that something is hard to use in some way, we could think how hard it is to use with disabilities. Bathroom is a good solid example, because we all use one. It is the basic necessity in any building or apartment. When it functions well nobody notices it. When it just does not function, everybody notices problems although they are not professionals on this matter.

This thesis was inspired by few recently constructed buildings. At that time when I started to plan this thesis, I did not pay too much attention to accessibility issues. Now it is all different and I spot every threshold, rail and staircase that are wrong in some way, accessibility wise. Bathroom or toilet rooms are one clear place to start observation whether the room or environment is easy-to-use or accessible.

Many new buildings have some recurring faults. It does not matter if building is built by a different constructor, this one thing almost always recurs. Toilet rooms main accessory is toilet paper dispenser. It is very often placed so it is really hard to use. As often it is also filled with hair thin paper that cuts in half with each pull. Accessing paper inside the paper dispenser, user often has to bend hand in weird positions. Usual mistakes are that toilet paper dispenser is placed too low or behind the water closet. Every time this same thing is very annoying but still it repeats in almost every public building.

Toilet paper dispenser’s placement would be an easy fix. Person that mounts it in the wall just would have to sit on the water closet, then think how the toilet paper

dispenser is easily accessible and then attach it to the wall. Very often public buildings have many separate toilet rooms or booths. These booths are very often narrow, so narrow that it would be impossible to even think that wheelchair or rollator would fit in that booth. That of course creates difficulty to fit chunky size toilet dispenser to small toilet room. It makes no sense that architect would want to fit four tight toilet room booths to space that can take only three. Those faults are most certainly done because there is a requirement to fit certain number of toilets in the building. The cost is then the functionality of that room.

In web this kind of design has gone the other around. Nowadays websites are designed to serve user the best possible way. That way website achieves more visitors and at the same time respecting company gets more revenue. About twenty years ago everything was done by rules by the system and users then just had to adapt to that. Today quality software, apps and websites are designed to serve rules by user not the systems. Maybe design of new buildings should look into this more. Good design makes all the difference. If good experience on a website will create more sales, good design inside of a building should do the same. In the end, place like toilet room is created for a person that is visiting that building and that is why it should always offer the best experience as possible.

2.4 ADA standards and Finnish ESKEH

To offer better user experience, design and accessibility, designer and architects must follow strict accessibility standards. Simulator is based on ADA and ESKEH standards. Simulation turns standards into tasks. Tasks contains environment testing like can toilet paper be accessed easily or can the toilet room itself be accessed at all. Simulation is like a gamified quiz form that asks questions like, how toilet paper dispenser should be placed to be accessible or how wide the toilet room should be. Instead of a quiz form simulation makes user to test the environment and see the faults clearly like, in a real life.

ADA bathroom guidelines for dummies is a good read. It shows simplified principles what to look for to inspect environments accessibility (Mobility.com, 2019).

A good starting point for accessibility testing is a simple rectangular room. This test room has no real-world counterpart, but it presents easily a starting point for accessibility testing in simulation. In game words this first simplified room is a tutorial that shows user the whole idea of the simulation and its testing features. Test room has a motorized door that opens with a button. Door controllers and placement of the doors are based on ESKEH standards.

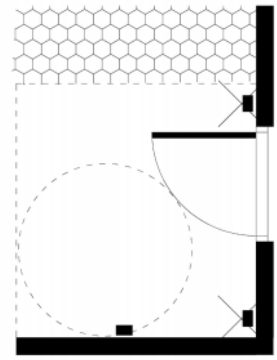

Image 2. Motorized door placement according to ESKE standards (Invalidi Liitto RY 2019d)

Making an environment based on Image 2 is a great starting point. It is simple to model in game environment and also the mission is just to get behind that motorized door.

Second tutorial is to use bathroom behind this motorized door. This is the main accessibility task trough out the whole simulation. Tutorial will teach user to spot different accessibility issues. These tutorials also will contain correct ESKEH based measurements that can be seen AR or augmented reality way in the virtual world. More about creating these tutorial environments to the game world in section 4.3.

2.5 Implementation to a real world

To test environment that exists in a real world it has to be first 3d-modeled in some way to turn it into a playable level. Currently there is no way to just start testing any environment. Buildings are built according blueprints. According to these same

blueprints even 3D-models can be created. With precise measurements that blueprint contain it is easy but time-consuming process to create a 3d model.
This should be automatized. This simulation realistically cannot lie on 3d-modeling that is made with human hours. That is too costly and inefficient. Later in this thesis there is more ideation on section 3.14 of what kind of tools could be used to scan existing environments.

2.6 Where and how simulation can be used

Any healthy person with no epilepsy can use simulation to test environments. Requirements are to have VR-capable computer and VR-kit. Currently mobile is not supported or lighter assemblies are not supported.

Anyone can download and use final product. Its source code will be also available on open source code sharing service the Github. Simulation is intended to give possibility test environments by anyone and share their thoughts and suggestions. Shared environments and comments do not bind people to just test their own local buildings, but they can also test shared environment on another side of the world. Shared feedback is one important part of this product. If the feedback of one particular accessibility issue is hot or more discussed it can be taken more seriously than one's word.

Social pressure is one way to push accessibility issues further. This simulation product needs a website, forum and social media as a support. Social media can easily promote issues to be taken seriously. As an example, if some public building would be missing an elevator it can be marked as an issue on a forum and social media. Hopefully, that would create a pressure to make that issue fixed. Forum and social media could work as a spokesman for people with disabilities. Forum and social media are a standalone feature among simulator, that gives anyone a change to participate in a work of making a world more accessible.

Creating a vivid community around accessibility testing-based simulation will be hard but plausible. Today's possibility to influence is much higher than in the past.

Today a single person's voice can shout louder than ever, with a power of social media. That is why it is so important to harness that power for a use of this product.

3 PLANNING THE SIMULATOR

3.1 Foreseen and known challenges

When starting this project, it was clear that it will not be easy to create all real life like features to this simulation. Starting from the natural movement of hand. Hand movement will not be like in a real world, but still it should feel natural or at least it should feel like it makes sense to just move hands in the air and move in the game while doing so. In the simulation wheelchair will move by waving hands in the air. Waiving simulates push of the rim of wheelchair. Careful planning of the movement is also crucial due to its effects to user's physical experience. VR is known to easily cause motion sickness. Motion sickness is a sea sickness like symptom (UX Planet, 2018). Overcoming motion sickness is the most important part in this project. If simulation causes motion sickness easily it is of course unusable at least for professional use. Motion sickness is also very user related problem and some people experience VR and its motion sickness effect stronger than the others. A study of University of Minnesota indicates that women more likely feel nausea when using VR than men. From the test group of women almost 80 percent felt sick after using VR glasses for 15 minutes. In the same study the general result was, that 56 percent men and women felt sick after 15 minutes use. (Bright Developers, 2017).

VR motion sickness is something that has to be taken seriously. Especially when in this projects case movement is made by waving hands rather than moving feet. Simulation's motion sickness effects can only be tested with test group in real life.

3.2 Overcoming the challenges

Good planning and testing are the most important things to overcome the challenge of VR motion sickness. It is impossible to tackle the whole problem for good. When I started to plan this thesis there were two main competitors on the market HTC Vive and Oculus Rift. Motion sickness is much caused by the equipment itself. VR kits like Vive and Rift contain some latency in the picture that they produce (Kjetil Raaen, Ivar Kjellmo. 2015). Latency means the time difference between the movement of the head and the time when movement appears inside the VR glasses screens. To understand latency better here are few examples what latency means. In multiplayer first person shooter (FPS) games latency has been a problem for a long time. Latency can make game unplayable. Especially latency is a problem in multiplayer games, game servers often even refuse player to join if the latency is too high. It is about the fairness against other players. Latency is created when player pushes a button on a keyboard or some other gaming device and the information then runs through cords to computer and via the internet to a server that can be hundreds of kilometers away. Player that pushed this button sees the movement almost instantly, but co-players do not. Other players will see the movement after the server has given that information to their computers and after their computer has processed that information. Latency is called lag in gaming world. Lots of lag can sometimes give unfair advantage or more often it just makes game unplayable. Lag in some games can be seen as game character jumping from one spot to another. In gaming world this is called warping and it is most often caused by a bad internet connection that causes lag. Warping in shooter games is an unfair advantage. It makes a character harder to aim or shoot at.

In VR kits latency is little bit different. It does not give any advantage to anyone, but it does create other problems. Latency in VR is created like follows. When players head moves the sensor on VR headset registers this movement. Movement info is then sent through cords to a computer and the computer creates or repositions image according to that movement. This little moment that the info is moving through cords to powerful computer and then after computer has processed that information and sent it to VR headset creates latency. Though that all happens literally on a blink of an eye, players eyes and brain are faster. Brain registers this movement faster than

sensors on a VR headset and at the same time presumes that image on the before eyes changes accordingly. This little moment though confuses brain and that's why latency is so bad thing when combined to VR.

VR kit also causes eye strain which is often felt soon after the VR glasses are put in the head. (Bright Developers, 2017). Devices like smartphones and computer screens can cause digital eye strain. Set of VR equipment is no exception although what causes digital eye strain with VR headset is little different than with smartphone's case. Staring smartphone screen for a long period of time can cause eye strain. How smartphones screen differs with VR is that there are two separate screens for each eye in VR headset. Smartphone screen is watched with both eyes focused to one screen. As a comparison eyes in VR headset are focused on two screens at the same time. This two-screen setup will create an illusion of stereoscopic image. The problem that causes eye strain is that unlike in real life, eyes watch almost the same spot whole time when using VR headset. VR headset create illusion of objects going to the distance and getting closer, but the reality is that eyes are staring two screens that only illustrate this change of distance. (EyeLux Optometry, 2019).

3.3 How fast brain is?

As a conclusion, tricking brain and eyes to think something that is not real is much harder than one may think. Brain is so fast that it spots singularities easily. To realize how fast brain really is it could be playfully compared to a computer. VR system is often attached to a powerful computer. To get a picture from a computer to a popular VR system like Oculus Rift HDMI cable is used as a connector. In Oculus Rift among a HDMI cable runs at least one USB cable. USB does not have anything to do with getting the picture from computer to VR headset. HDMI is what transfers the image data to those two separate screens on the VR headset. Continuing this playful comparison, one must know HDMI cables maximum data transfer rate. As in Oculus Rift's case its HDMI cable is version 1.3 or 1.4 that has a maximum data rate of 8.16 Gbits/s (Wikipedia 2019c). Brain is a lot faster. Brain transfers 2328306,44 Gbits of information every second (Quora, 2017). As said, this is just a playful comparison and it is nothing too scientific. It still shows that we have a super-fast computer in

our head. This may be one reason why it is so hard to fool and when fooled it causes nausea and other symptoms.

3.4 Selecting tools

In planning stage there were two potential game engines to choose from. One was Unreal Engine and the other was Unity. I chose Unreal Engine to do this simulation with. Other choice was to choose from two quite similar systems HTC Vive and Oculus Rift. Somehow, the Oculus Rift is a system that I like more, it is bit easier to set up. Also, the Rift's controllers are better for my hand. Both VR systems have full support in Unreal Engine and Unity so that did not affect to a decision of choosing Oculus Rift for being the test tool. Notable is that if a game is made and tested with Oculus Rift it is almost certainly compatible to be run also on HTC Vive. Controller configuration is the only thing that in this case may need some adjusting. Adjustment can be as simple as question in-game, that asks what VR system you are using. Choice will then affect to controller configuration.

3.5 About Unity and Unreal

Unity and Unreal are both game engines that have built-in features like physics engine, 2d- and 3d-game support and potential to publish games for smartphone and tablet platforms like Android and iOS. Unity was first published on year 2005 and Unreal Engine was created for game named Unreal in 1998. Unity was originally created to be an exclusive game engine for Mac OS X. At the time of writing this Unity supports already 25 different platforms not just Mac OS X.

There are some factors that affect to which game engine to choose. Both game engines are free to use and downloadable. At the stage of publishing both engines start to require some kind of payment. Unity has a proprietary license and has free and paid versions. Unreal on the other hand does not have paid version at all. Cost of Unity depends on how much company creates money in a year. Small companies that generate less than 100.000 dollars per year can use Unity for free. Same goes for personal use (Wikipedia 2019d).

Four is the current version of Unreal Engine. Originally on 2014 when Unreal Engine 4 was published, developers had to pay monthly fee to be able to use this software. Fee was reasonable 19 dollars per month per subscription. The plan was to get more developers to develop with Unreal Engine instead of Unity. After one year at 2015 Unreal was made available for free with all its features (Pc Gamer, 2015). At this stage Unity had still the Pro version. Pro version contained more features that the free version. Soon after Unreal announced that it will make all subscriptions free Unity made all features available for free version also.

Using and developing are different than publishing the final product. Publishing a game with either game engine has a cost if game sells enough. If a game made with Unreal engine makes more than 3000 dollars per quarter, company responsible has to pay 5 percent royalty of the revenue that game has generated. Unity in the other hand is much cheaper to use when game is published depending on how big the development team is. Unity costs 125 dollars per user per month and it does not have financial eligibility like Unreal Engines royalty system does. (Unity Store, 2019)

3.6 References made with Unity and Unreal

Rust is a survival game / shooter made with Unity. Rust is available for PC, Linux and Mac. Rust is a good example of a game that is made with Unity and looks just astonishing. In some circles Unity is known to be a game engine for mobile gaming or games that are little lightweight. Rust is nothing like that. Rust has huge environment to explore and is a multiplayer game, also as already mentioned graphics are very good (Rock Paper Shotgun, 2014).

Wasteland 2 is RPG that is based on 1988 published game Wasteland. Game was back in the 1988 highly popular and praised. Game sets in post-apocalyptic era. Game is point and click based game and camera perspective is two and a half dimensional. Games idea is to survive and accomplish missions (Pc Games N, 2015).

From mobile side there are many games made with Unity like Deus Ex: The Fall, Assassin's Greed: Identity (Soomla. 2015)

3.7 Serious games that run on Unity

Serious games are not as popular as games that have been created for entertaining purposes. It is quite hard to check what game engine specific game engine is made with. With block buster games it is much easier to check the technology behind those games. Big popular games most often have Wikipedia page where the technology specifications can be checked.

Satakunnan ammattikorkeakoulu Oy (SAMK) has made few serious games using Unity. One of those games is ski jumping game. Game has three different stages and it is controlled with a chair. Chair has pressure sensors that are used in the game to steer and for jumping. Chair has 9 sensors and Arduino Mega as a brain of the chair. Data is transferred via Bluetooth to the game. Game can be played with a normal tablet or smart phone, but it has been designed to be used with 65-inch Android mega-tablet Yeti. Game makes player to jump up from the chair to jump in ski jump level, lift each side of bottom to steer in snowboarding level and throw balls towards the tablet in the last level. Game is for rehabilitation, exercise and for fun. Main focus group was elderly people, but game is suitable and fun for anyone.

At the time of writing this thesis, one more serious game is under development at SAMK. Team of researchers are creating a game for Yeti-tablet that is aimed to be playable even with most demanding users. Test group is from Paloaho school at Kuopio. Test group contains lots of different disabilities and one of those is autism. Some of pupils do not communicate at all and some communicate with sign language. Test group also has a majority of pupils that can't read.

All this creates a lot of demands for a serious game that is produced to that group. The game is developed with Unity. Game is very simple and does not contain any tutorials. Game starts with blank canvas that player can draw on by tapping certain spot on the screen or by drawing with hand or some other pointing device. An airplane

then flies to that spot and leaves a steam like trail behind it. If there is a cloud on that path planes trail turns in to the same color as that cloud is. Idea is that player figures out color change while playing the game and then starts to use it to create patterns or other artistic features. Game also teaches colors in sign language. There is a sign language sign as a picture that means some color name. There is also that same color as respective colored figure under that sign language picture. In-game that active color must be flown trough to get next color. This is also learned by just playing the game. The game is easy to create and publish with Unity for mobile platforms like Yeti-tablet. Game can also be played with lighter VR system like Oculus Go or Google cardboard. VR gives advantage to make game more accessible and makes game to be playable even without hands. VR version of this game works just by looking certain spot instead of touching a screen. From rehabilitation perspective the VR version of this game rehabilitates neck spasms.

3.8 Why I chose Unreal?

Unreal 4 engine is used by big companies like Bluehole that developed very popular multiplayer game PlayerUnknown's Battlegrounds (PUBG) (PUBG Corporation. 2019). Another one very popular multiplayer game named Fortnite Battle Royale is also developed using Unreal 4 engine (Epic Games.2019).

Unreal and Epic Games have very good documentation and lot of videos in Youtube. Videos and tutorials can be found from channels like Tesla Dev, Ben Ormstad and Unreal Engine. Those create an easy access to start game development. My own opinion is that Unreal looks a lot better than Unity. At least when the starting point of a game has been created Unreal looks just astonishing. User interface of unreal is much more appealing than Unity's.

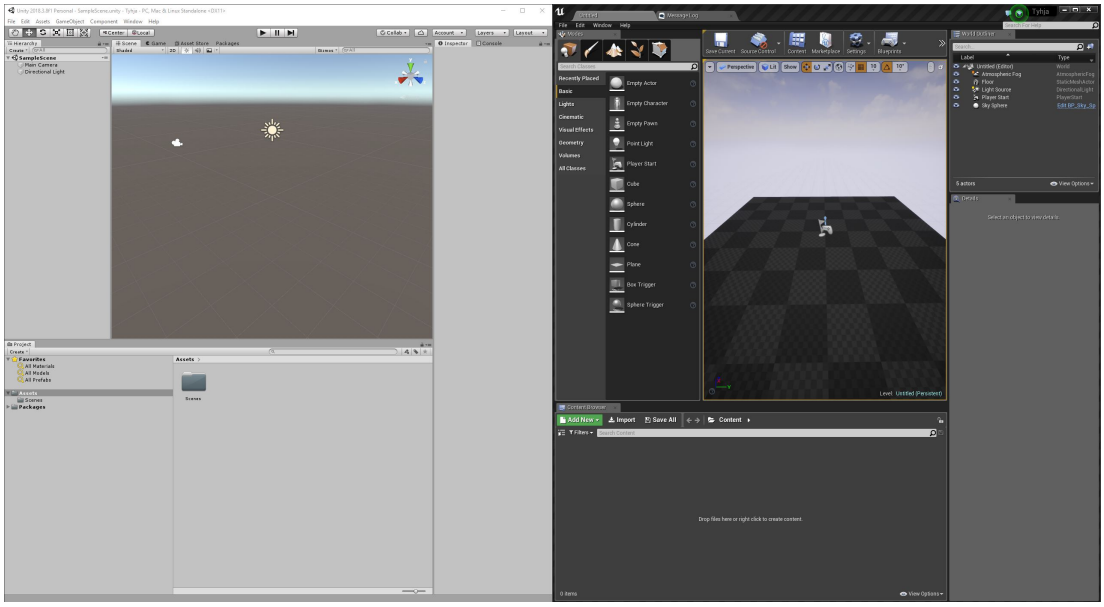

Image 3. Side by side comparison of Unity's (left hand side) and Unreal's user interface when empty project is established.

Unreal has icons that will gives some idea of what type of object can be used in the game. Plain game objects like sphere can also be added to game environment by dragging them. In Unity adding a plain game object is added from a dropdown menu. Both works great when one gets familiar with the system. For a starter who has no idea about game objects, Unreal is maybe somewhat easier to start with compared to Unity. Unreal Engines user interface sort of suggests objects or at least they are visible when Unity will not give any hints about how one starts building the game environment.

Another factor is that Unity requires a lot of plugins to work like Unreal. Unreal has many built-in features. At the time of creating the game for this thesis, effect called particle effect was one feature that was missing from Unity, or at least it was not too extensive. Particle effect is not important for this game but at the time I developed a first person shooter (FPS) game as a separate project and that required all kinds of effects like particle effect.

Depending on the usage, plugins are Unity's advantage. Features like Bluetooth compatibility was totally missing from Unreal when this project started. Bluetooth is not important for this game, but it clearly demonstrates that these both engines have their own advantages. At the time of writing this thesis Unreal has gotten its own

Bluetooth plugin made by TahaHISHRI. Unfortunately, this plugin is targeted only for Android platform, so Apple and other devices are out of question. Plugin is free though.
As a conclusion Unity is much better choice for demoing. There is a lot of free plugins that support all kinds of platforms. Unreal Engines plugins are slowly evolving, but at this time Unity is better option.

Last reason why I selected Unreal was node based coding system that is built-in to Unreal. Unity also has node based coding systems, but the difference is that those are not built-in. Unity requires a plugin like Playmaker to be installed before any visual scripting can be made. At the time of writing this Playmaker plugins normal price was 65€. More about Node based coding in section 3.10.

3.9 How game world differs from a real world and how it affects development

Unlike real world, in game world every object can defy physics. Every object can float in the air without any support. Any structure or level can hold any amount of weight. In game world almost anything is possible. Image 4. illustrates Unreal Engines levels starting point. Everything in the image 4. is on a platform that is floating in the air. Although it is unrealistic, the platform can hold any number of chairs, tables or anything else despite it is floating in the air without any support. That just is, how game engines work before developer tells objects to work differently.

Game-character can throw an object that weights a ton in the air without any effort. That object can then travel as long it wants to direction it was thrown at. That would be totally unrealistic in real world. The harder part in game world would be to get that object to hit ground as realistically as possible.

In game world every object needs guidelines or rules to function like the objects they were intended. Game objects need information what those objects are. Object like a normal drinking glass needs physics, weight and even rules how it will break in the pieces if it hits the ground. Ground itself also needs rules that how it will behave when something touches it or hits it. Ground could have rules like how soft or hard it

is and what kind of sound it will make when glass hits it or how it will alter the glasses hitting sound.

There are so many things that must be considered before getting a simple real-world action to feel like it is real. Fortunately, algorithms like physics do not have to be coded by the developer anymore. Game engines like Unity and Unreal have very powerful built-in physics engines.

As mentioned, every object can float in the air without any support. Normally how building a game world starts is that there is a platform that is floating in space. Then other objects can be piled up on top of it. In Unity and Unreal objects like ground need rules to make object interact correctly with collision. Collision means moment when object hits another object. If collision rules are not applied correctly other objects will just go through other object like ground.

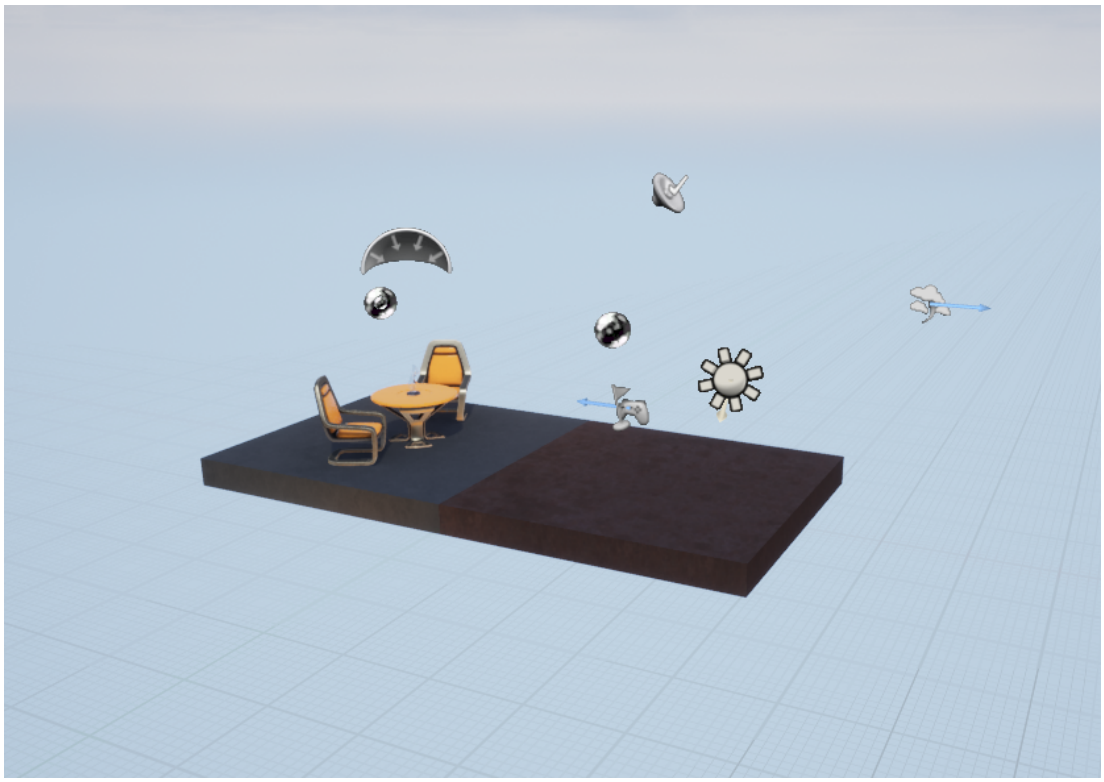

Image 4. Illustration of starting view of empty project in Unreal Engine 4

3.10 Node based coding

Lego robots use Mindstorm EV3 software. EV3 is a node-based coding software. It uses blocks to achieve certain tasks. Basically, each block is predefined to do something like for example run motor forward or backward (Joe Olayvar & Evelyn Lindberg. 2015).

How to use these blocks is up to user. Blocks usually are chained to each other. In block chain the first block can have a rule like, how long does it take robot to start using motor to move forward. Second block could be that robot moves forward as long as it detects color like green on the ground. Legos Mindstorm sets have great features and parts like optical sensors. Lego Mindstorm optical sensors can be used as wanted. Parts have predefined coding counterparts as a block in the coding side that make using parts really easy.

To understand better blocks and node-based coding one must understand a little bit of how coding works. IF-statement is fundamental piece of coding. Coding is roughly asking question whether something is or is not and according to that do something. Below three different ways are exhibited to do an IF statement. Every sample is from a different program but still they all do the same thing (Image 5.).

The difference of these three ways in different programs is obvious. But for a person that is familiar with coding can see the similarities. These three do exactly the same thing. In this case all examples below use or may use boolean type variable in comparison.

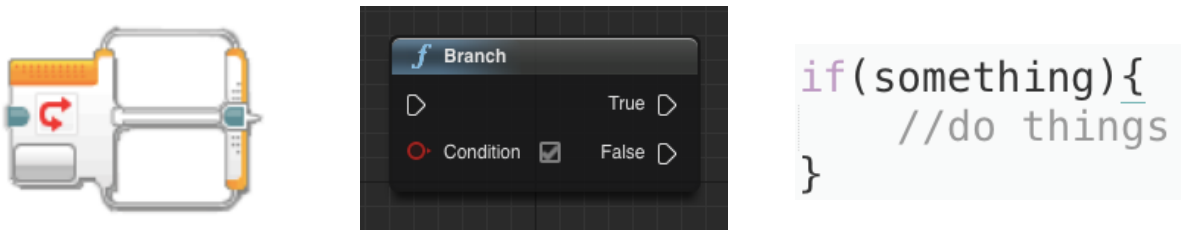

Image 5. If statement in three different programming softwares. Starting from left Mindstorm EV3, Unreal Engine 4 and Javascript. (Source: docs.unrealengine.com, lego.com)

The original reason to choose Unreal as game engine is the native node-based coding system. I was little familiar with C++, but not enough to start coding with it. C++ is the language that Unreal Engine uses on the background even though visually coding can be made with node-based system called Blueprint. Above in Image 5. there is a simple boolean comparison with three different systems or languages. Nodes in Unreal Engine are objects that contain everything that is needed for coding. Every node has its own functionality, but they are used commonly. Nodes then can be used in graphs to make almost anything without writing a single line of code.

Unreal Engine has system called Blueprint. Every object in the game can have its single or most usual case multiple Blueprints. Blueprints are also capable to communicate with each other. Communication between Blueprints makes it easy to create custom code for some objects if needed and makes code more organized. Building a game's functionality with nodes in Unreal Engines Blueprints is pretty much drag and drop based building.

Blueprints are very intelligent and powerful. When combining nodes system is actively checking compatibility of nodes. System won't usually give user a chance to combine wrong nodes together (Image 6.). System will warn about the problem with an error message that tells the reason why two nodes can't be combined. (Epic Games, 2019.) This is also one advantage of Blueprints or node-based coding against pure coding. Notable though is that also programs like Microsoft's Visual Studio will also warn, if some variable is in compatible. Warnings are much like in Unreal Engine, but those often will not count the whole case like Unreal Engine's Blueprint does and due to that will give much more room to errors.

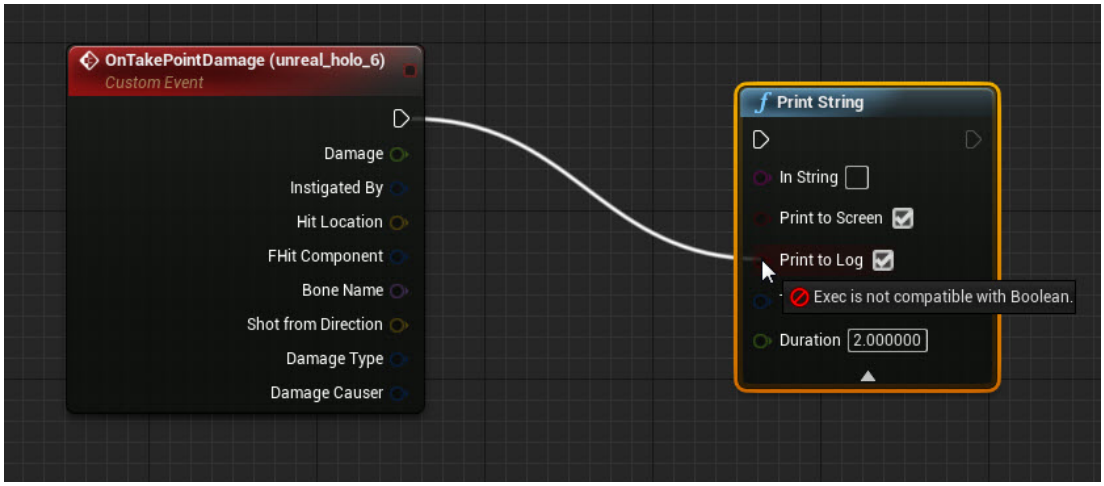

Image 6. Illustration of two incompatible nodes in Unreal Engine (Epic Games 2019)

It is clear that node-based coding has helped a lot producing this product to demo stage. Youtube has very good tutorials about making components and functionality for almost any game. Epic Games is providing lots of support also. They also have their own Youtube channel that has very good and extensive tutorials. These tutorials make starting the development with Unreal Engine easy.

3.11 VR / Virtual Reality and Serious VR-games

At the time of writing this thesis VR-systems like HTC Vive and Oculus Rift have been on the market for quite some time. There is a lot of games, solutions and serious games to choose from. From educational perspective one free game is a great example why VR is so important for this project. This case is all about the visual learning that is the most powerful way to learn (eLearning Industry, 2017).

The best serious VR-game that I have come across with so far is Sharecare VR and it is available at Oculus store for free. Game allows user to explore human body closely. Sharecare VR has human organs modeled very precisely and user can even dive into a modeled heart to see how blood flows in practice. It is quite incredible how an engineer now knows how the valves of heart work and how these valves get stressed from atherosclerosis. The best thing is that not only I know how the heart valves work in theory, but VR experience has also left me a visual mark to my brain that is like video about how those valves work and block up. It is totally new

effective way to learn things. Below there is a comment about Sharecare VR from a nurse and midwife who wraps the point of why to implement VR to educational software.

Image 7. A screenshot from Oculus stores Sharecare VR page. (Oculus.com. 2019.)

Implementing user to environment is really important. Hands on experience is important when intending to teach more than just a theory (Education Week Teacher, 2009).

Although, VR is not a real “hands on” experience it is perception of being physically in non-real world. When we experience things like changing heart rate or atherosclerosis building up in front of our eyes leaves users neurons with a feeling of heightened sensation (Wikipedia 2019f).

For other purposes VR’s possibilities are endless. Interesting study was made in 2012. Study was about body representation where an illusion of having a virtual body was studied to understand how brain processes this non-real body. Results were that with virtual reality body stimulates both skeletal muscles and brain computer interface (BCI). (Llobera, González-Franco, Perez-Marcos, Valls-Solé, Slater & Sanchez-Vives, 2012.)

These above mentioned discoveries show that serious games will benefit over VR a lot. Impressiveness is very important to make non-real as close to a real-world case as possible. This is why this thesis's product has a VR capability. Instead of playing or using simulator through a normal PC-monitor it helps user to feel and implement to non-real environment before it even exists. All inaccessible faults can be tested beforehand. When testing with enough deep immersion, experience should leave memory stamp to a user's mind. Noticeable part is that user will test environment from wheelchair. This setup of testing a building, street or other environment is impossible to create with any other way, than virtual reality. With a help of neurons of human brain and their experience, creating more accessible buildings and environments may be easier in the future from designers and architects view of point (Llobera et al. 2012).

3.12 VR motion sickness

It may sound somewhat funny to have motion sickness with VR-glasses. After the first use of VR set it is easy to understand how easy it is to get almost seasickness-like feeling. Roller coaster is quite easy understood example, because almost everyone knows what roller coaster feels like in the real-world. In virtual reality roller coaster feels real almost like it would be happening before user's eyes. Brain will think that user's body is in motion although user is sitting still on a chair. It does not matter if graphics or environment in the game world are not photorealistic, brain still feels that movement is happening. On roller coaster game it is easy to get that freefall feeling that one would experience on a real roller coaster when the roller coaster train is going suddenly downwards.

As it is great advantage from developer's perspective to be able to give physical feelings to player, there also a big downside. Feeling of that funny freefall feeling means that brain has interpreted visual feedback of VR-glasses as a sudden drop. That happens on a stand still chair without any movement.

Small test groups in SAMK have tested some of roller coaster games with Oculus and HTC Vive equipment. These test groups have not complaint about motion sickness feeling although they are moving very fast in virtual environment while in SAMK's test lab they are in reality sitting still. This may be because we have prior expectations of how roller coaster works.

Simulator's wheelchair and roller coaster train are very similar in a VR-world. Simulator's wheelchair like also roller coaster train move in the game-world although user's chair is not physically moving. This makes a conflict between what brain expects and what is actually happening. Brain interprets visual feedback from VR-glasses as motion, but at the same time body does not feel like it is physically moving (Wikipedia 2019g). This creates illusion of self-motion, that may lead to seasickness like feeling or even physical response (National Center for Biotechnology Information, U.S. National Library of Medicine. 2019).

3.13 Moving the wheelchair

To overcome virtual reality sickness, natural movement needs to be added to the equation. Simulation of course could be used with VR-glasses and keyboard, but that combination would produce exact problem that is described before. Simulation is used with controllers that comes with HTC Vive and Oculus. As simulation is about testing an environment with a wheelchair the controllers are used to mimic the motion of pushing the rims of wheelchair. Pushing is not intended to be totally realistic because it would create some foreseen difficulties from usability point of view.

The image 8. shows how movement is planned to be produced in the simulation. Left hand side picture with red arrows describes how a real wheel could be pushed forward. Although it is a realistic way it means that VR-controller should follow this path to accomplish movement. This though is not the most optimal way to produce motion.

Right hand side picture describes how the movement is produced in the simulation. Blue arrows present the path that controller has to follow. Green angle mark means that the angle of this path can be changed in the simulation's settings to get it suited for anyone. Purple gradient color presents offset. Offset is also adjustable from the settings. Offset will help to achieve the mimicked rim push feedback to the system. Widen or in game world heighten touchable area in 3d-environment will be easier to follow than narrow one. Adjustability makes it easy to adjust it to anyone's needs. These arrow paths and offset can be also seen inside the simulation to help to see the path that is needed to be touched.

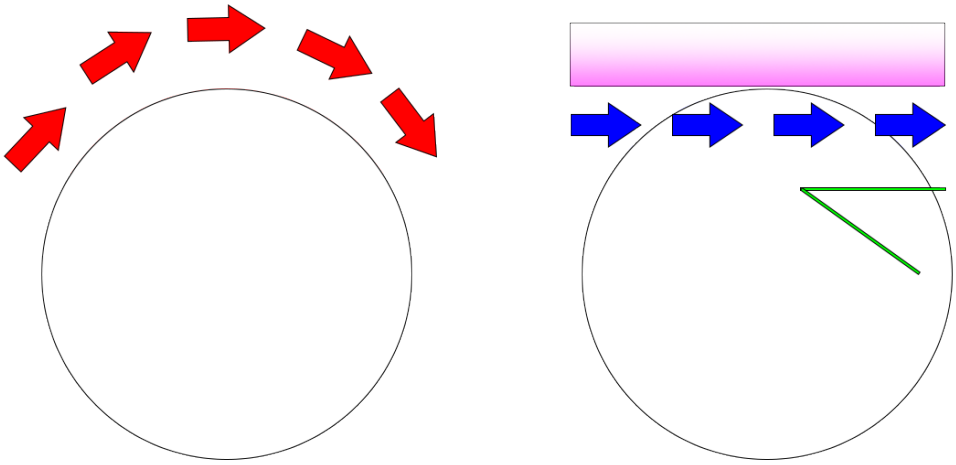

Image 8. Illustration of plan how to create movement path for the simulation

When controller enters the push area it will make a small haptic feedback to notify the player that player's hand has entered the area that interacts with the environment. When the rim is virtually pushed haptic feedback is little more tense. More about haptic feedback in section 4.

To push the rim few things has to happen. In this case HTC Vive controller is the example controller because it has a grip button. In Oculus Rift's case instead of grip button, the button under forefinger can be used. Pressing the grip button mimics a squeeze of the rim. When controller is inside of the purple area and grip button is

squeezed player can see the movement in the game. By moving hands forward and letting the grip go in the end point makes wheelchair to move in the game-world.

Turning is rather similar to moving back or forward. By entering the purple area and by griping left- or right-side player can then turn the wheelchair. Turning may also be done by pulling left and right side to different ways.

3.14 Scanning the environment to get testable 3D-model

There are few 3d-scanning apps for smartphones. One is Trnio. Trnio uses smartphones camera to scan object front of it. Any object can be scanned by moving around the object. Object may even be human. Trnio also texturizes the object with the image that was pictured during the scan. Unfortunate feature of Trnio is that it is only for iOS platform. Other rather same kind of app is Qlone. Qlone has almost exact features compared to Trnio. Qlone is available for both Android and iOS platforms. Both are very useful and powerful tools to catch 3d-modeled image from an object (Image 9.).

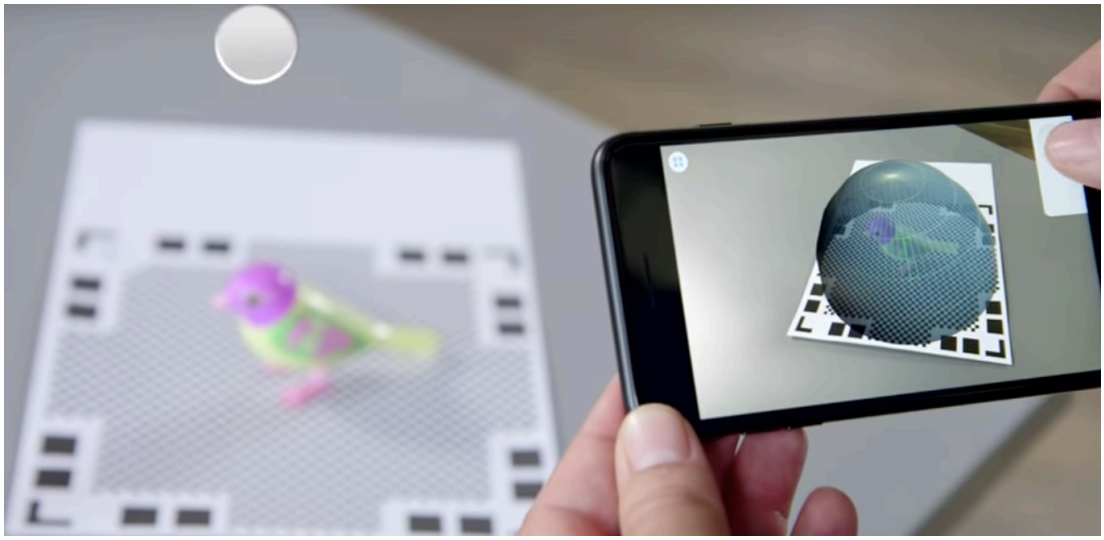

Image 9. Qlone scan in progress (from https://www.qlone.pro/ Youtube embed)

This all may seem like a dream case considering how easy it is to turn real-world object to 3d-modeled objects in just few seconds. All real-world objects can now be scanned with a smartphone and then transferred into a game world. This is partially right but far from the reality.

These apps produce 3d-models that are super realistic looking. From game point of view super realistic is more often bad thing than good one. Computer to render realistic objects requires a lot more power than rendering simple objects, like cubes or planes.

It all is about polygons or complexity of polygon mesh particularly. Polygon mesh normally is like a solid object that is created from small triangles. These triangles are called polygons. Note that polygons are not always necessarily triangles (Wikipedia 2019h). Blender is very popular 3d-modeling software that is totally free to use. Blender uses polygons to render objects. Polygons one side or line is called edge and flat area is called face. Polygons corners can be considered as dots and those are called vertices (Wikipedia 2019i). Maybe the simplest object in 3d-modeling is a cube. Although it has only six sides polygons can make it more complex than that. Polygons may divide simple cube to many different faces. Below there is an illustration from Blender that shows how simple cubes polygons form objects.

Complexity of an object is important to understand in this matter. The more complex the object, the more it requires resources from a rendering computer. Comparison (Image 10.) shows that the same looking object can be lot different from polygon perspective. The first cube has only 12 polygons in it. In comparison, the last one with most squares has 768 polygons in it. As mentioned more polygons more power hungry the object is.

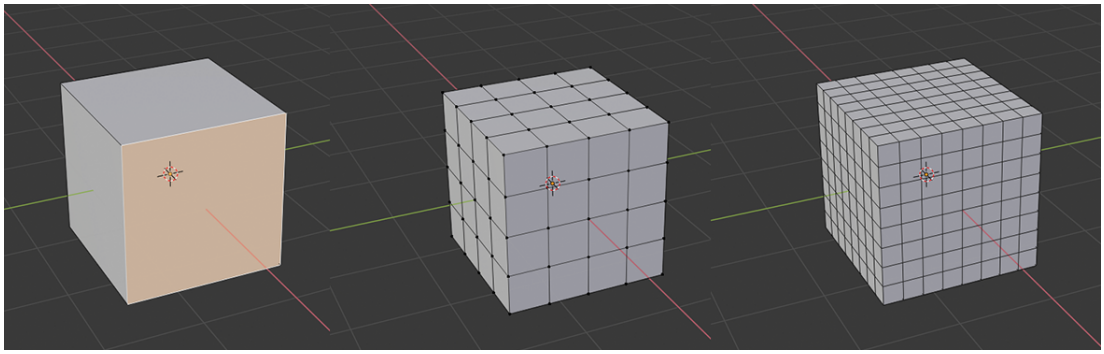

Image 10. Comparison of simple cubes polygon count in Blender.

Comparison like this may seem insignificant and useless when only one object is examined. In game world there are thousands of game objects and in that case polygon counts start to matter. Nowadays computers have huge calculating powers. With a good hardware it may not matter much how dense the polygon mesh is at start. When game-environment and the game itself turn complex enough, the performance issues start to arise. This is why it is so important to keep low polygon counts as high in a priority list as possible. Especially in mobile devices, performance issues will be clear in very early stage. In mobile it is crucial to optimize objects that are used in the game or solution. Why mobile has to be considered in matter is that Oculus Go is comparable to a smartphone and therefore has limited performance. At time of writing this thesis, Oculus Go does not have traction of hand movement but in the future, there might be controllers for both hand in Oculus Go. Oculus Go in that case would be perfect platform for this simulation to run on.

Now that polygons are explained, 3d-scanned models can be understood better. It is almost certain that every object that is scanned will be a power-hungry object. Therefore, without optimization those objects cannot be used straight from the app.

Another problem with these scanning apps like Trnio and Qlone is that they are designed to scan objects, not environments. Without that feature those apps cannot solve the issue where 3d-environments should be easily scanned and transferred into a testable game level.

Currently, there is no cost-effective way to scan environments. Laser 3d-scanner attached to a drone could work. Although it would have exact same issues as those previously mentioned smartphone apps. By scanning environment with laser would definitely be effective, but it is almost certain that doors and small closures like toilet rooms can’t be scanned accurately enough. Scanner would not make a difference between a wall and a closed door. Making environment work with this technique would not work without postproduction. All the doors and badly scanned areas and objects need to be modeled by hand. Objects like soap or toilet paper dispensers and door handles would have to be modeled into the level no matter how well the environment

is scanned. It may be even faster to just model the whole environment manually than to fixing 3d-scanned model to work.

There is no solution that would just transfer an existing real-world environment into a testable simulation level. That is very unfortunate because it means that simulations like this will require some or lots of money and work hours to make real-world environments accurate. Hopefully future would and probably will bring solution to this problem. For now, the most cost-efficient way to 3d-model environments for this kind of simulation is to follow closely blueprints and possibly existing 3d-models of the real-world environment.

4 PRODUCTION AND TESTING

4.1 Creating tutorial environments

In this section, creation of environments is shown as pictures. Created environments are very simple and created mainly for illustration purposes. Unity game-engine was used as a tool to create a fictional environment. Unity was driven with Macbook Air that had latest MacOS installed. Running Unity with Macbook Air that has relatively low hardware specification is a great example, that how little performance Unity requires to be run on and used for developing purposes. For a comparison Unreal Engine can be installed on same device but it is unusable. Macbook Air that was used to create these demo environments had 1,8 Ghz Intel Core i5, 8gb of DDR3 random access memory (RAM) and built-in graphics card Intel HD Graphics 6000. As a developer I personally would never suggest anyone to purchase a computer with this low hardware specifications. In some point computer like this will have performance issues, especially when publishing demos, test versions or the final product.

All objects are created inside Unity and those objects are very simplified. At this point there is no need to use modeling software’s like Blender to create complex models of buttons and doors. Models like realistic looking buttons or door can be

placed afterwards after environment is ready and tested. This section covers only modeling but not the functionality coding of the objects.

Modeling in this case started from a simple rectangular room. Room has one door that is motorized. Below there are illustrations that show how the modeling of this room could be done. Tutorial room should have at least list of objects as follows. Floor and ceiling are planes. Walls are cubes and for this purpose 5 cubes are needed. One more cube is to present a door and in model it has a pink color (image 11.)

Plane is an object that is two dimensional and does not have any thickness. Cube on the other hand is an object that has also depth. Plane is a good object to be used as wall, roof or floor that are not intended to be looked from behind or side. Plane is much more cost efficient on performance view of point compared to a cube because it consists of 4 vertices instead of 8 vertices. (Wikipedia 2019b)

Model's two walls are shaped as too long on purpose. This same model can be also used to present a hallway. Only one wall in a middle need to be moved farther and model turns into a hallway instead presenting only a room (image 11.).

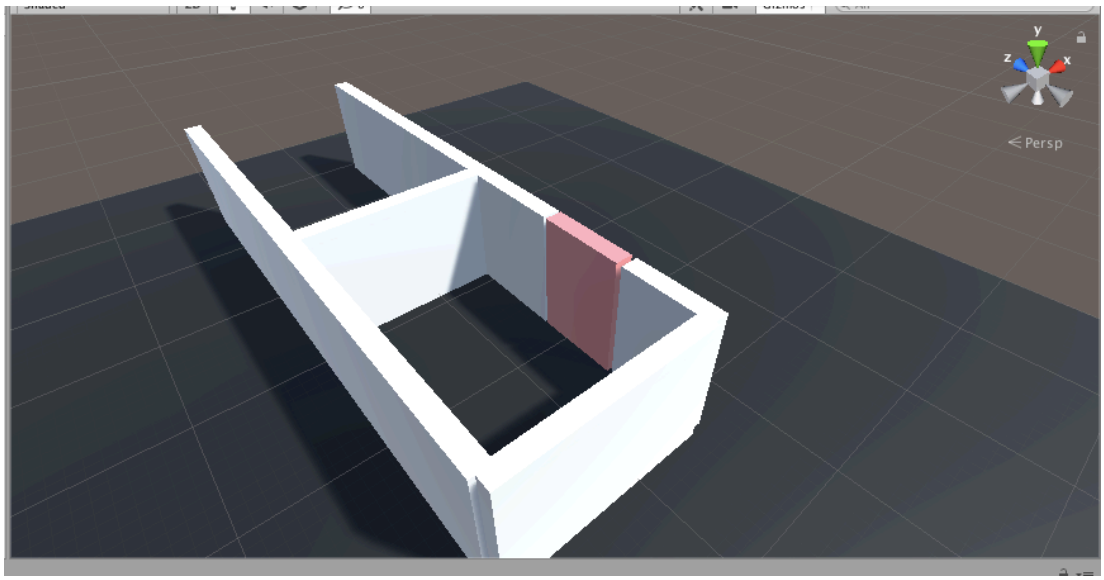

Image 11. Presentation if a simple room in Unity

Next room needs a door opening buttons to present wrong placement and correct placement of the door controller. Wrong and correct placement of the door controller knobs are based on ESKEH standards.

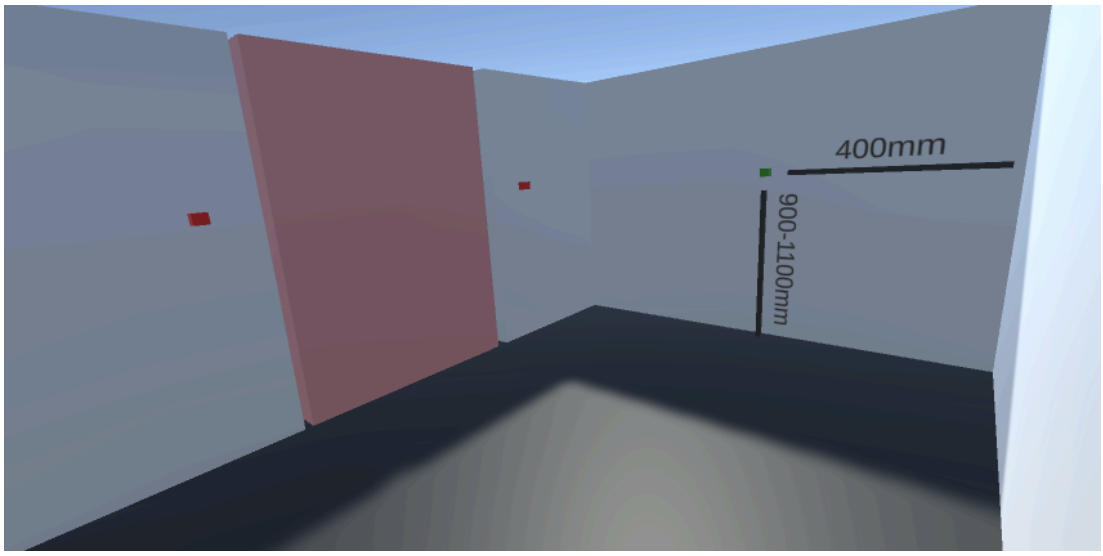

Image 12. Correct and wrong placement of door controller knobs with legend of correct measurements.

Tutorial's part one is now modeled. Now a simple toilet room can be added behind this motorized door. Room needs water closet, sink and foldable arm rests and those are placed according to ESKEH standards. Simple cubes present these toilet rooms accessories.

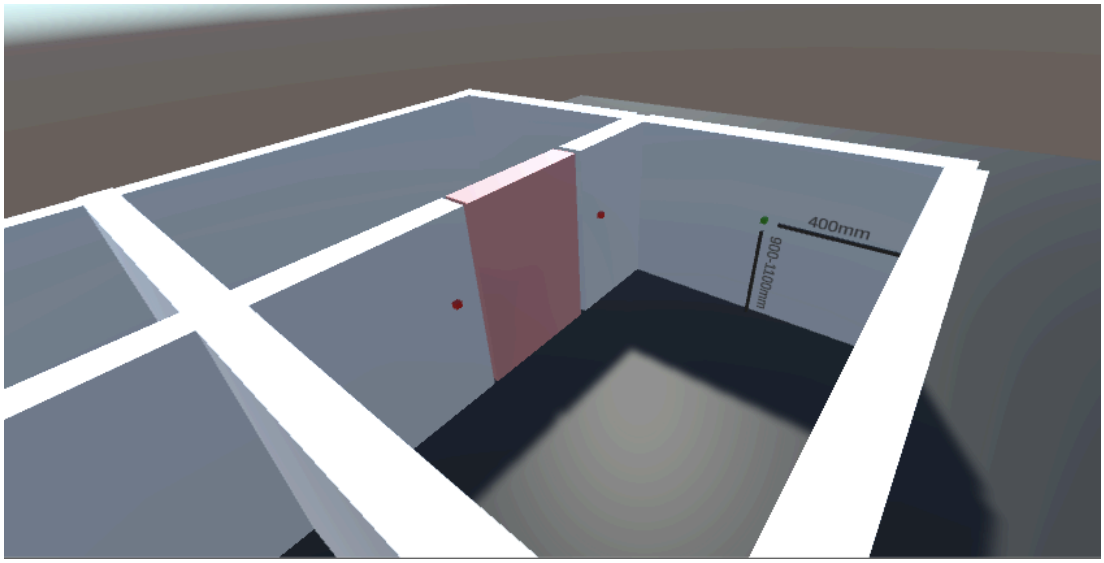

Image 13. Toilet room is added behind the motorized room.

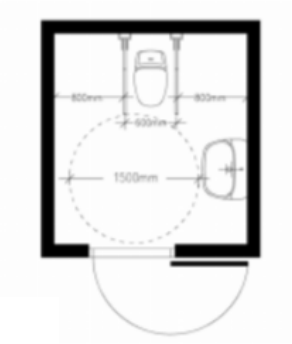

Image 14. ESKEH toilet room standards (Invalidiliitto Ry. 2019)

Image 14 is created as a mirror. Measurements to be met are as follows. 800mm free space on both sides of water closet and it has to extend from wall to water closets back line by 200 – 300mm to make sure that foldable arm rests are not on the way when stepping from wheelchair on to water closet. As standard recommends water closet could be installed 200 – 300mm from the wall to enable this.

Arm rests should be installed 600mm apart from each other's center point. In case that arm rest are not foldable their installation height should be 750 – 800mm. In this tutorial model height of those arm rest is 800mm and they are foldable.

There has to be 1500mm area of free space on front of sink and water closet. (Invalidi Liitto RY 2019a)

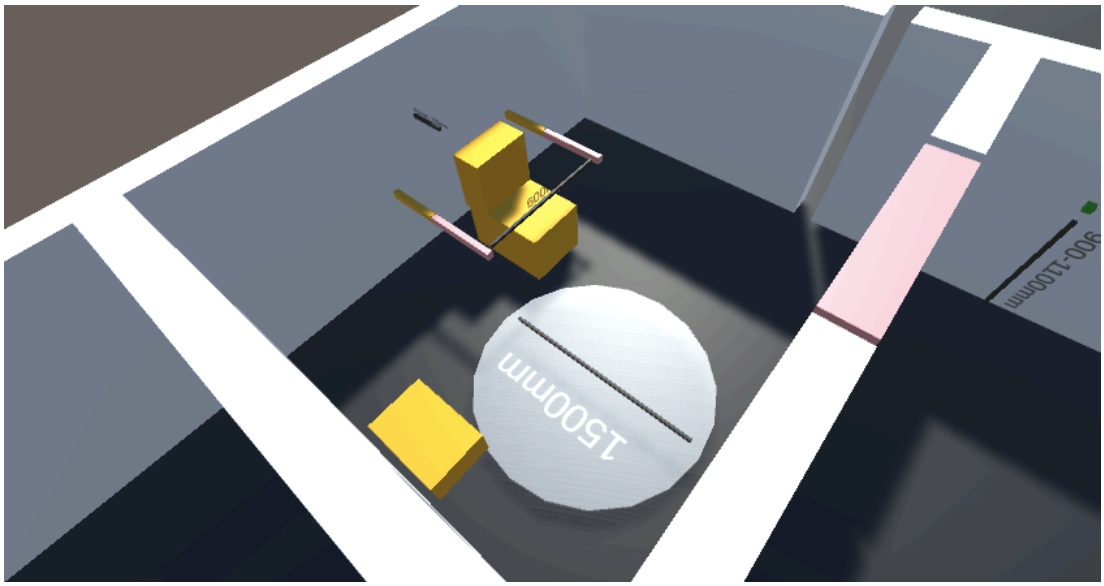

Image 15. Final toilet room model according to ESKEH standards.

Game has augmented reality mode or AR-mode, that may be enabled in-game. AR-mode can be turned on or off from the settings. AR-mode is intended to guide user towards understanding the correct measurements of accessible design better. AR-mode measurements can be found on image 13, 14 and 15.

4.2 Coding the functionality

Environment is fully static before it has some built-in functionalities. Door opening buttons will not do anything before some coding is done first. Although there is wrongly and correctly placed door buttons those all can still have the same functionality. Functionality is easily transferred to all objects that need push-like feature. In real-life, button usually is pushed in before it does its functions that it is intended to perform. 3d-modeled button does not understand pushing neither it does not move in wall when touched. These all features need to be coded before those can be used in the game or simulation.

Buttons in this test environment are very simple. Buttons need feature to know when they are touched and make some small movement when touched. Life-like movement of the button is extra feature that is not priority. On priority order, touch is the number one feature that button need to have. Interaction may be done in test environment easier than in real-life. Button may understand touch before it is even physically touched in VR-world. This is done by adding collision box over the button. Collision box is special object that is intended to use to detect if object goes through it, this is called touching in-game. In Unity square collision box is called Box Collider.

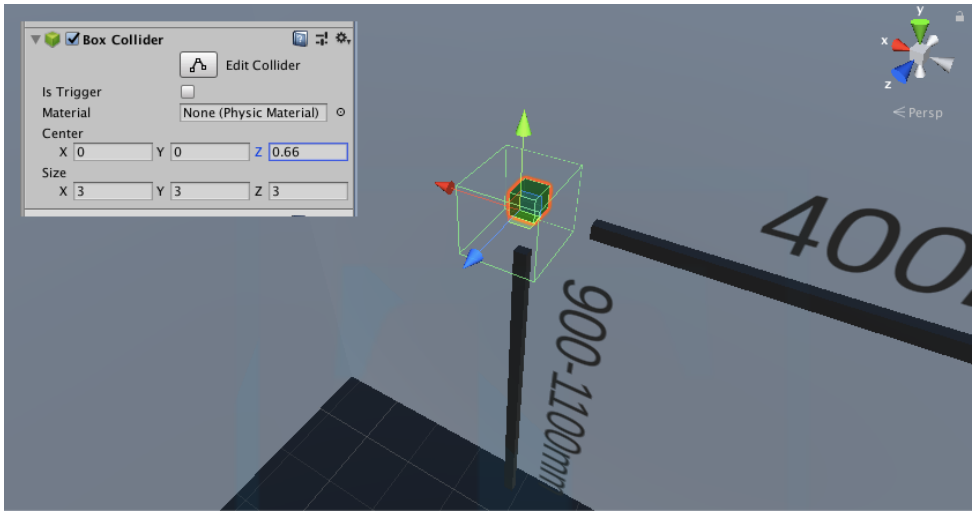

Image 16. Box collider in Unity

Box collider is three times bigger than the button to make it easily touchable. Code part for touch is quite simple. Button object needs code, that may be called touch.cs. In touch.cs file touch is triggered with code below.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class touch : MonoBehaviour
{
    public GameObject DoorToOpen;
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        if (Input.GetMouseButtonDown(0))
        {
            RaycastHit hit;
            Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);

            if (Physics.Raycast(ray, out hit, 100.0f))
            {
                print(hit.transform.gameObject.tag);
                if (hit.transform.gameObject.tag == "nappi")
                {
                    DoorToOpen.GetComponent<opening>().open = true;
                }

            }
        }
    }

}
```
Image 17. Touch.cs script

Code above uses mouse click as a trigger. Raycast gives object that was touched with mouse and stores that in “hit” variable. The most inner if-statement then checks if

“hitted” game object contains tag “nappi”. If it does code then changes doors boolean variable that contains info if door should be closed or open. DoorToOpen game object must be set in Unity to match the door that is wanted to be triggered by this button.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class opening : MonoBehaviour
{
    float openTime = 0;
    public float doorOpenAngle = 90.0f;
    public bool open = false;
    float defaultRotationAngle;
    float currentRotationAngle;

    public float openSpeed = 1.1f;

    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        if (openTime < 1)
        {
            openTime += Time.deltaTime * openSpeed;
        }
        transform.localEulerAngles = new Vector3(
            transform.localEulerAngles.x,
            Mathf.LerpAngle(
                currentRotationAngle,
                defaultRotationAngle + (open ? doorOpenAngle : 0),
                openTime
            ),
            transform.localEulerAngles.z
        );
    }
}
```

Image 18. Opening.cs script

Script in image 18. illustrates code that is attached to door that is opened. Boolean “open” is that variable that previous script touch.cs did change. Basically, the door just waits for open-variable to change and when it changes it gives code permission to start turning the door towards the angle that is stored in variable named “doorOpenAngle” that has a default value of 90.

Above exhibited codes are examples of Unity-made codes. Those two scripts are just a small piece of the whole system that lies under the bonnet of big simulation project. Example codes and scene were made with Unity because scripts of the Unity game engine are much easier to be explained than complex Blueprints that Unreal uses.

Blueprints also would have been screenshots instead of a real code and that way little harder to be explained on detailed level.

4.3 Accessibility cases in this project

In games usually when tutorial part is over game itself begins. Like a normal game, this product will contain missions that are accessibility cases. Missions are tasks that user needs to do to complete the level. Level in this case may be place like small coffee shop. Below there is description of a simple mission with submissions and what it takes to make mission work from logical and coding view of point.

How to go inside a building where the front door is closed but not locked?
Getting into a building could be the first mission. In this case there are few things that have to be considered in gamifying. Open doors would be too easy so there must be some obstacles instead of easy access into the building. Obstacles like doors will prevent user from entering the building. User needs to pull the door to get it open. Door way may have double doors, that means both doors need to be open at the same to fit a wheelchair. In demo scene doors were coded to be self-closing doors. That meant that it is not enough to pull just one door open and enter. First user must pull one door open and then jam it with wheelchairs corner. Then second door may be opened, and user can pull the wheelchair in.

This simple task is somewhat demanding to create. It requires good collision detection from doors and wheelchair. Also, collision detections should not be able to be overridden. This may easily happen if self-closing script does not understand that door is touching some object that should jam the door like in a real world. If these two objects and their scripts do not communicate well between each other, door may go through the wheelchair. This kind of bugs need to be evaded. Bugs destroy the realism of the simulation (Image 19.).

Image 19. Illustration of child object not having its own collision detection (Bohemia Interactive. 2018)

How to get into a toilet room?

This second mission contains few submissions that are crucial or optional to complete. The first submission is to find the toilet room door from the building. As gamified features there could be non-playing character (NPC) that can be used as a guide. NPC could give directions to toilet room if asked. Asking help from NPC would give a score to user. As games today are not so strict, other possibility would be that user just goes and finds the toilet room without guidance. On the way player can notice and point guides towards toilet room if there is any. User will gain score from all found guide texts.

Opening the toilet room door

The first mission was to access the building trough a double door. Level could contain toilet room that has a normal door and a toilet room that has a door that can be opened from a button on a wall. Either can be used to complete the mission. Using non-accessible toilet room has the same problems that the front door had. To open the door it needs to be pulled at the same time as trying to enter it. This particular task needs coordination of both hands. Other VR-controller needs to be attached to a handle of the door and other hand has to ably some reversing motion for the wheelchair or otherwise the wheelchair would block the way of the moving door. This demonstrates well the real-world situation of this same event and its accessibility issues.

How to get on to a toilet seat

To get on to a toilet seat user must be able to drive wheelchair somewhere beside the toilet seat. If that is not possible it can be marked as an accessibility issue. At the same time user will gain score for spotting a problem in a building. Illustration image 18. below shows the space requirement for the wheelchair. Correct measurements according ADA-standards can be seen on image 20. If the toilet seat is inaccessible, user can still try to access it from another angle.

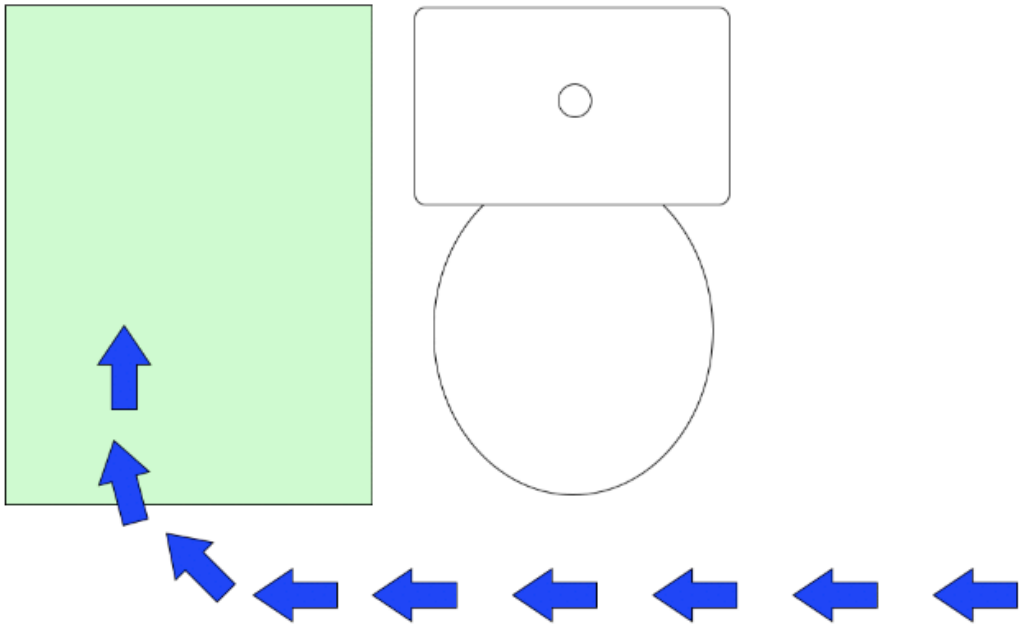

Image 20. Illustration of the space needed beside the toilet seat to fit a wheelchair

Other missions

After getting on to the toilet seat, it is time to test how easy it is to get paper out of the toilet paper dispenser, if it is even in reach. Here is a list of other missions that could be found on this same building: Washing hands, using soap dispenser, using hand towel dispenser, getting out of the toilet room.

Simulation should contain several simple and normal real-worl

how hard it is to move inside a building that is not accessible. As comparison the simulation should also contain levels or buildings that are fully designed to be accessible. Simulation's gamified features are all about gaining points out of ordinary

things like washing hands. Gaining points is aimed to give user a stronger gamified feeling like games normally do. Although tasks are easy and points are gained easily, success and achievements are not the point of this game. Game will give points for spotting accessibility issues although it may seem that points were gained from completing a task, like hand wash. In the end of every mission the player will get a full report of all accessibility issues. Report will contain spotted issues and the issues that were overtaken by the user although those were accessibility issues.

4.3.1 Sound design

Sound is very important factor in games. Quiet environment just does not feel like it would be alive. This is why it is very important to add sound and sound effects in to game world. Places like shopping center should have that hustle and bustle like ambient noise on a background to get immediate feeling of a live environment. Even if eyes closed that environment feels like a shopping center. Sound is crucial part of the experience that player will gain from playing games (Amplifon, 2019).

Doors and knobs do have to make some kind of sound when used. Elevators, radios floor under wheelchair and lamps on a roof all should make some kind of real-world like sounds to implement user deeply into a game-world. Different surfaces like wooden or concrete floor should create different sounds when player moves on them. Rugs on a floor should alter floor's original sound depending on what surface it is on. Creating realistic gentle sound while rolling with wheelchair creates much more immersive feeling of movement. Sound is important factor in immersion or at least it is often considered as one factor. Still there is no theory that would proof that immersion and sound have a strong connection. This is because it is hard to proof what makes a game immersive. Immersion happens unconsciously and it is hard to point out which factors are most important and what are less important (Sander Huiberts, 2010).

As one example, Zombies, Run! is a mobile game that is little unconventional when compared to any other zombie or survival game. It uses only audio to tell its story. Zombies are created in players mind and not on smartphones screen. Idea of the

game is that, player is always on the move and doing missions. Missions are quite simple ones. Usually, player just has to run somewhere and pickup something from some place. In between missions the game now and then alerts about zombies, after that player simply must run to escape. There are no zombies behind the player in the real world but when running in a dark forest it certainly starts to feel like zombies are close. This all is created with sounds and sounds only. At least with good imagination serious game like Zombies, Run! feels very immersive.

4.4 Haptic response and implementation

Before getting into a matter of why haptic response is so important for this simulator, one must first understand what haptic response is. Haptic response is basically some kind of response from virtual environment to a real world. Haptic technology is also called as kinesthetic communication and comes from Greek word haptikos. Hapticos(άπτικός) means “pertaining to the sense of touch”.

In smartphones there is usually haptic response option as default. Cellular phones have existed for quite some time, and vibrations feature is almost as, old feature as, cellular phone itself. Although vibration feature has existed for quite some time, the smartphone era created a new need for vibration feature in smartphones. Haptic response in smartphones helps to write text with virtual keyboard. When virtual key of a virtual keyboard is pressed, it gives a tangible small vibrating feedback that can be interpreted as, key press. What usually gives that sense of vibration is a small motor or motors that inside the device. One example of this kind of motor is inside of LG Optimus L7 II (Wikipedia 2019a).

Very common solution of haptic response in games is a force feedback system in racing games. It requires a dedicated steering wheel and pedals to get this response. In racing games, haptic response is not as saddle as feedback when virtual key is pressed. Gaming steering wheel with force feedback feature will give really strong resistance when driving car to corners. This on its own tries to give player a feeling of being behind a steering wheel in race car (Orozco, Silva, El Saddik, Petriu. 2012).

It would be great to have steering-wheel-like feedback when trying to push wheelchair forward. This though would require two wheels on both sides, that are size of wheelchair rims. These two steering wheels then would have to be customized for this particular use. This would require lots of coding and testing to get wheels resistance and feedback perfect. One considerable thing is that those steering wheels are not intended for this use. It may be even impossible to adapt this kind of product to this purpose.

Using customized equipment would not fill one purpose of this simulator. Simulator is intended to be used with any VR-equipment or desktop computer that is powerful enough. Any customized peripherals will decrease markets and add costs to the end product. The easier the implementation to clients existing equipment, the more market it has.

4.4.1 Haptic response in VR

Touch is the most primitive sense that we humans have. We are able to understand the world around us just with the sense of touch. Hands belong to more sensitive parts in the human body (Roope Raisamo & Jukka Raisamo, 2011). Haptic response is an important part of bringing the sense of touch from virtual world to real world. Of course, it is nothing compared to real touch but currently at least on widely available commercial products like HTC Vive and Oculus Rift, it is the only way to create the sense of touch.

Virtual reality devices, like HTC Vive and Oculus rift, both have haptic response feature. HTC Vive and Oculus rift both have one haptic motor inside of each controller. This motor spreads the haptic feedback to whole controller (iFixit, 2016). These motors can then be programmed to do certain tasks. As described above this motoric response is important when interacting virtual objects such as virtual keyboards. In virtual world these controller's haptic motors can be controlled trough a code. Normal and most common application is to add haptic feedback when a controller touches a virtual object. It is totally up to a developer innovativeness that how haptic response is used. One case could be instead of touching of an object, player could

feel a vibration when entering to forbidden area inside the game such as radioactive area. It is totally up to a solution how haptic response is used (NotionTheory/Medium.com, 2017).

In this simulator the haptic response will give a feeling of a touch of an object. Important feedbacks are touching the wheelchairs rim and touching an object to interact inside the simulation. In this simulator, lack of haptic response can be used to guide user to understand that certain game object does not interact, or at least it is not intended to interact. To make clearer that certain objects are un-interactable, a sound and/or visual response can be added to object. Objects that are interactable can make clear major key sound and un-interactable objects minor-key sound. Using simple major (happy) and minor key (sad) sounds should give a user a feeling that something is forbidden, or something is allowed (NME, 2013). More about importance of sound in Sound design section 4.3.1.

5 CONCLUSION

5.1 What was accomplished

Simulation did not get completed during this thesis. Lots of theory and good practices have now been gathered. During the long time period of making this thesis has also given some advantages. Technology has evolved again few steps. Resolution in VR-devices has improved; controllers and base stations have evolved and mobile standalone devices like Oculus Go, have entered the market. As mentioned previously, in this thesis Oculus Go may be very suitable device to run this kind of simulations. The only downside currently is that it does not track players movement except the head.

This thesis can be considered as a preliminary requirement analysis for an accessibility simulator. Over the thesis process, a bunch of demos has been generated to test the parts of the simulation in practice. Most of programming was made with Unreal but some code was made also with Unity. Codes and worlds of these two competing

game engines can't be combined but both have proven to be valid tools for developing this kind of a simulation. The Unreal version of the uncompleted demo simulation is available for anyone on Github.

From hardware point of view both HTC Vive and Oculus Rift are good choices to run this kind of a simulation. The final product must support at least these two platforms.

5.2 Human Resources

Almost from the beginning, it was clear that this is a huge project. It simply is not one man's job. It would be plausible if one would do this full time. That was simply not the case. While digging deeper into this thesis, it was also very important to do the research right rather than just start developing. Development of a simulation got really far. It has huge environment with some interactive features, like doors and a door man that player can act with (Image 21.). Wheelchair was also programmed in the way that it could be used in the future. Wheelchair still needs to be programmed to be used with VR kit, currently it only works with a keyboard. Reason for this is that it is much easier to test functions and features of the environment with regular keyboard and mouse setup. The simulation has some finished gamified features like scores and animations.

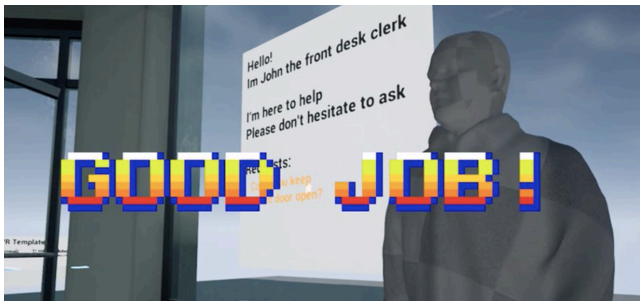

Image 20. Gamified features of the game.

Making these features that are lower on the priority list at the middle of the development process was a clear mistake. Development should have been planned better.

During the development, I created a task list, but while one task was completed many others had been added after that. Developing this simulator just for the thesis purposes was overwhelming. Thesis itself should be created also. At this point I started to pay attention more to theory side of this project. Theory and studies behind this project are very interesting. At this point it is clear that when this simulator is finally created, it is much better than it would have been in the beginning. I now have much more theory, studies and understanding for accessibility to back up the simulator.

5.3 Faced problems

The biggest problem was that the project was simply too big for one developer. Among normal working hours plus development at home it was soon clear that time and motivation are the biggest issues for development. For this big project it is now clear that planning, designing and development simply take lot of time. Especially development is huge time consumer. For a simple collision detection development time can be 4 to 8 hours. To make much more complicated collision detection that detects collision from at least 4 points will add more hours to that. The challenge is to make those collision point to work together. As simple case what collision detection should do, is to detect collision from one point, and if another point is also collided the code should prevent character from moving to that direction. Otherwise character will move through a wall. Collision detection is explained better on production and testing part.

5.4 Finalizing the product

To make this product complete, it should have one realistic finalized level like shopping center that has gamified interactive features. To get it as real as possible the shopping center should be based on a real shopping center like Kauppakeskus Puuvilla or Isokarhu that are located in the city of Pori. Model needs precise copy of doors, stairs and thresholds. Modeling is not enough because doors, stairs, knobs and lifts also need programmed functions. Shopping center does not need to be photorealistic. Main purpose of the model is to test accessibility issues. If this tool is used as a test tool time, effort and money for gaining photorealistic experience is not its main

purpose. As explained earlier, it is very important to have these test sessions as realistic experiences as possible. Graphics, sounds and implementation are important things, but they do not have to be exact. Main thing is to have convincing experience that is memorable.

5.5 Future development

At Samk there is a big demand for this kind of a product. I have already gained a lot of interest for this kind of a product inside Samk. Producing simulator like this need its own funded project. With a team of skilled programmer's simulations first demo should not take more than half a year to get finished.

Demo would solely concentrate on the testing tool part. As previously mentioned, an idea is to have two separate products, a gamified product for common people and a professional testing tool for architects and designers.

5.6 Marketing the final product

Social media id widely used today. Still, I personally do not think that it is right the channel to market the test tool part of this product. This simulation is much more than just a game. It is a tool and probably not too interesting for a regular person to spot on Instagram or Facebook feed. Regular consumer is not the target market for this simulation. This is more for corporations that are involved in accessibility design and development. To get word out, maybe traditional press like, medical press-releases would be good place to promote this product.

Main goal would be to test environments before they are built. Also testing different renovation scenarios can be tested before big investments so the real estate press releases should be good place to promote this product. It may be far-fetched but I would hope that this simulation would be available for everyone for free. Main idea of the whole product is to make accessibility and its issues known better. Secondly that professionals can test the buildings to make them more accessible. Although the product would be free, it would still need constant investments. The final product, of

course, is not ready when it is released. It will contain bugs and systems like HTC Vive will change and upgrade in the future and will create incompatibility issues. To fix these issues and make any updates, product needs a team of programmers also in the future. Invaliidiliitto Ry or other similar of organization would be a great supporter of this product. With an existing network that they already have the marketing would be much easier than the starting of the whole marketing process from the beginning.

REFERENCE LIST

Amplifon. 2019.The psychology of sound in video games.
Referred: 10.3.2019
http://www.amplifon.ie/resources/playing-with-your-mind/

ARA. 2014. Asutko sinä hissittömässä kerrostalossa?.
Referred: 16.3.2019
https://www.ara.fi/fi-FI/Ohjelmat/Hissiinfi

Bright Developers. 2017. Why Motion Sickness Happens in Virtual Reality.
Referred: 27.7.2019
https://www.brightdevelopers.com/motion-sickness-happens-virtual-reality/

Education Week Teacher. 2009. Teaching Secrets: How to Maximize Hands-On Learning.
Referred: 24.3.2019
https://www.edweek.org/tm/articles/2009/09/02/tln_cody.html

eLearning Industry. 2017. Visual Learning: 6 Reasons Why Visuals Are The Most Powerful Aspect Of eLearning.
Referred: 24.3.2019
https://elearningindustry.com/visual-learning-6-reasons-visuals-powerful-aspect-elearning

Epic Games. 2019a. Nodes.
Referred: 24.3.2019
https://docs.unrealengine.com/en-us/Engine/Blueprints/UserGuide/Nodes

Epic Games. 2019b. Fortnite Battle Royale.
Referred: 21.9.2019
https://www.epicgames.com/fortnite/en-US/play-now/battle-royale

EyeLux Optometry. 2019. Virtual Reality and Digital Eye Strain.
Referred: 29.7.2019
https://www.eyeluxoptometry.com/news/virtual-reality-digital-eye-strain/

Finlex. 2004. Ympäristöministeriön asetus esteettömästä rakennuksesta.
Referred: 10.7.2019
https://www.finlex.fi/data/normit/28203-F1su2005.pdf

Harbor City Supply. 2019. Small or Single Public Restrooms | ADA Guidelines.
Referred: 23.4.2019
https://www.harborcitysupply.com/blog/small-or-single-public-restrooms-ada-guidelines/

iFixit. 2016. Oculus Touch Teardown.
Referred: 17.3.2019
https://www.ifixit.com/Teardown/Oculus+Touch+Teardown/75109

Invalidi Liitto RY 2019a. ESKE kartoitusopas.
Referred: 27.6.2019
https://drive.google.com/file/d/1498DMnSPbcBhVXk4LJUbP7qwsu59kITW/view

Invalidi Liitto RY 2019b. Kulkuväylä.
Referred: 10.7.2019
https://www.invalidiliitto.fi/esteettomyys/julkinen-rakennus/kulkuvayla

Invalidi Liitto RY 2019c. Historia.
Referred: 16.3.2019
https://www.invalidiliitto.fi/invalidiliitto/organisaatio/historia

Invalidiliitto RY 2019d. Esteettömyyskartoitusopas 2019.
Referred: 15.8.2019
https://drive.google.com/file/d/1498DMnSPbcBhVXk4LJUbP7qwsu59kITW/view

Joe Olayvar & Evelyn Lindberg. 2016. LEGO Mindstorms EV3 Programming Basics.
Referred: 22.9.2019
https://www.sos.wa.gov/_assets/library/libraries/projects/youthservices/legomindstormsev3programmingbasics.pdf

Kadettikunta Ry. 2014. Veteraanien perintö – itsenäinen isänmaa, Sotainvalidit.
Referred: 16.3.2019
http://www.veteraanienperinto.fi/vepe/index.php/fi/ryhmia/sotainvalidit/index.html

Kjetil Raaen, Ivar Kjellmo. 2015. Measuring Latency in Virtual Reality Systems.
Referred: 20.9.2019
https://hal.inria.fr/hal-01758473/document

Llobera, González-Franco, Perez-Marcos, Valls-Solé, Slater & Sanchez-Vives. 2012. Virtual reality for assessment of patients suffering chronic pain: a case study.
Referred: 26.9.2019
https://doi.org/10.1007/s00221-012-3352-9

National Center for Biotechnology Information, U.S. National Library of Medicine. 2015. More than a cool illusion? Functional significance of self-motion illusion (circular vection) for perspective switches.
Referred: 14.4.2019
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4531211/

NME. 2013. The Science Of Music – Why Do Songs In A Minor Key Sound Sad?
Referred: 17.3.2019
https://www.nme.com/blogs/nme-blogs/the-science-of-music-why-do-songs-in-a-minor-key-sound-sad-760215

NotionTheory/Medium.com. 2019. Inside of a Vive Controller.
Referred: 17.3.2019
https://medium.com/@notiontheory/inside-of-a-vive-controller-7c9de18ac9d3

Occupational Safety and Health authority in Finland. 2019. Non-discrimination.
Referred: 16.3.2019
https://www.tyosuojelu.fi/web/en/employment-relationship/non-discrimination

Oculus.com. 2019. Sharecare VR.
Referred: 27.3.2019
https://www.oculus.com/experiences/rift/1656800021020362/

Orozco, Silva, El Saddik, Petriu. 2012. The Role of Haptics in Games.
Referred: 17.3.2019
https://www.researchgate.net/publication/221923248_The_Role_of_Haptics_in_Games

Pc Gamer. 2019. Unreal Engine 4 is now free.
Referred: 29.7.2019
https://www.pcgamer.com/unreal-engine-4-is-now-free/

Pc Games N. 2015. InXile to give Wasteland 2 a graphical overhaul with Unity 5 engine.
Referred: 21.9.2019
https://www.pcgamesn.com/wasteland-2/inxile-to-give-wasteland-2-a-graphical-overhaul-with-unity-5-engine

PUBG Corporation. 2019
Referred: 21.9.2019
https://www.pubg.com/

Quora. 2017. How fast does the brain think?.
Referred: 29.7.2019
https://www.quora.com/How-fast-does-the-brain-think

Rock Paper Shotgun. 2014. Impressions: Rust's New Version.
Referred: 21.9.2019
https://www.rockpapershotgun.com/2014/11/26/rust-new-version-review/

Sander Huiberts. 2010. Captivating Sound,
Referred: 15.9.2019
https://download.captivatingsound.com/Sander_Huiberts_CaptivatingSound.pdf

Soomla. 2015. Top 10 Unity Games Ever Made.
Referred: 21.9.2019
https://blog.soomla.com/2015/01/top-10-unity-games-ever-made.html

Statista. 2019. Virtual Reality (VR) - Statistics & Facts.
Referred: 19.9.2019
https://www.statista.com/topics/2532/virtual-reality-vr/

Thinknum Alternative Data. 2018. Consumer interest in VR is declining according to sales data trends.

Referred: 19.9.2019
https://media.thinknum.com/articles/sales-data-shows-that-consumer-interest-in-vr-is-waning/

Unity Store. 2019
Referred: 29.7.2019
https://store.unity.com/

UX Planet. 2018. Motion sickness in VR.
Referred: 27.7.2019
https://uxplanet.org/motion-sickness-in-vr-3fa8a78216e2

Wikipedia 2019a. Haptic Technology.
Referred: 17.3.2019
https://en.wikipedia.org/wiki/Haptic_technology

Wikipedia 2019b. 3D-projection.
Referred: 15.8.2019
https://en.wikipedia.org/wiki/3D_projection

Wikipedia 2019c. HDMI.
Referred: 29.7.2019
https://en.wikipedia.org/wiki/HDMI

Wikipedia 2019d. Unity (game engine).
Referred: 29.7.2019
https://en.wikipedia.org/wiki/Unity_(game_engine)

Wikipedia 2019e. Unreal Engine.
Referred: 29.7.2019
https://en.wikipedia.org/wiki/Unreal_Engine

Wikipedia 2019f. Immersion (virtual reality).
Referred: 24.3.2019
https://en.wikipedia.org/wiki/Immersion_(virtual_reality)

Wikipedia 2019g. Virtual reality sickness.
Referred: 14.4.2019
https://en.wikipedia.org/wiki/Virtual_reality_sickness

Wikipedia 2019h. Polygon (Computer Graphics).
Referred: 14.4.2019
https://en.wikipedia.org/wiki/Polygon_(computer_graphics)

Wikipedia 2019i. Polygon Mesh. 2019
Referred: 14.4.2019
https://en.wikipedia.org/wiki/Polygon_mesh