Eemeli Heinonen

HoloLens research and demo application development

Metropolia University of Applied Sciences

Bachelor of Engineering

Information and Communication Technology

Thesis

7 March 2018

Metropolia

Author Title Number of Pages Date	Eemeli Heinonen HoloLens research and demo application development 35 pages + 2 appendices 7 March 2018
Degree	Bachelor of Engineering
Degree Programme	Information and Communication Technology
Professional Major	Mobile Solutions
Instructors	Heikki Laitala, Passionate Software Developer Petri Vesikivi, Principal Lecturer

The thesis was made as a research and development project for a Finnish software development company. The goal of the final year project was to create an in-depth documentation of HoloLens' capabilities to be used as a knowledge bank by the company's personnel and customers, and to develop demo applications that could be used interactively to demonstrate the device.

The main focus of the thesis is Microsoft's HoloLens augmented reality (AR) head mounted display (HMD). To get a better understanding of the topics covered, the study presents an introduction to augmented reality which also covers history of AR, some recent use cases and an overview of several manufacturers' HMDs.

HoloLens has two HD resolution screens and a field of view of 30-degrees. The device's performance is comparable to a high-end mobile device and the battery life is 2–3 hours in active use. HoloLens also has 12 sensors that it uses to interact with the environment among other things.

The thesis also covers the development process of two demo applications that were made for the final year project. The development of the first application covers the implementation of each of HoloLens' interaction methods one by one along with testing of text and image recognition and the process of making a simple 3D object more engaging and interactive. The second development process was only two weeks long and relied heavily on image recognition with Vuforia SDK.

The research revealed that AR is a promising technology that is very likely to become common once it matures and becomes more accessible. As for now, the AR HMDs still have technical limitations that restrict them from becoming popular among consumers.

One of the demo applications was presented to a customer during the development phase and the thesis itself is available for the company's personnel as a source of information on AR HMDs.

Keywords	Augmented reality, AR, HoloLens, Unity, Vuforia, image recognition

Tekijä Otsikko Sivumäärä Aika	Eemeli Heinonen HoloLens-tutkimus ja demosovellusten kehitys 35 sivua + 2 liitettä 7.3.2018
Tutkinto	Insinööri (AMK)
Tutkinto-ohjelma	Tieto- ja viestintätekniikka
Ammatillinen pääaine	Mobile Solutions
Ohjaajat	Passionate Software Developer Heikki Laitala Yliopettaja Petri Vesikivi

Insinöörityö tehtiin ohjelmistokehityksen asiantutijayritykselle tutkimus- ja kehitysprojektina. Työn tavoitteena oli tehdä kattava dokumentaatio lisätyn todellisuuden laseista toimimaan tiedonlähteenä yrityksen työntekijöille ja asiakkaille sekä kehittää demosovelluksia laitteen esittelyä varten.

Työssä tutustuttiin lisätyn todellisuuden historiaan, selvitettiin tämän teknologian kypsyyttä ja perehdyttiin sen toteutustapoihin. Insinöörityössä keskityttiin pääosin HoloLens-laseihin, mutta myös muita lisätyn todellisuuden laseja tarkasteltiin vertailun vuoksi.

HoloLensissä on kaksi HD-resoluution näyttöä 30 asteen katselukulmalla, sen suorituskyky vastaa tehokasta mobiililaitetta ja laitteen akunkesto on noin 2–3 tuntia aktiivisessa käytössä. HoloLens sisältää myös 12 sensoria, joita laite hyödyntää muun muassa ympäristön kanssa vuorovaikutukseen.

Insinöörityössä tehtiin myös kaksi demosovellusta. Ensimmäisen sovelluksen kehityksessä sovellukseen lisättiin jokainen HoloLensin vuorovaikutusmenetelmä sekä kuvan- ja tekstintunnistus. Toisen demosovelluksen kehitystyö oli lyhyempi, vain noin kahden viikon pituinen, ja se keskittyi vahvasti kuvantunnistukseen. Demosovellusten kehitykseen käytettiin Unity-pelimoottoria, sen Vuforia-laajennusta ja Visual Studiota.

Insinöörityö paljasti lisätyn todellisuuden olevan lupaava teknologia, joka hyvin todennäköisesti yleistyy, kun se kypsyy ja tulee helpommin saavutettavaksi. Tämänhetkisillä lisätyn todellisuuden laseilla on teknisiä rajoituksia, jotka estävät niiden yleistymisen kuluttajien käytössä.

Insinöörityössä tehtyä demosovellusta esiteltiin asiakkaalle jo sovelluksen kehitysvaiheessa, ja itse työ on hyödynnettävissä lisätyn todellisuuden lasien tietolähteenä tilaajayrityksen työntekijöille.

Avainsanat	AR, HoloLens, lisätty todellisuus, Unity, Vuforia, kuvantunnistus

Contents

Metropolia

List of Abbreviations

AR	Augmented reality. A technology where digital content is added into the users' field of view.
VR	Virtual reality. An immersive technology where the user puts on a headset to replace their vision with 3D content.
UWP	Universal Windows Platform. A Cross-platform development environment from Microsoft.
MVP	Minimum viable product. Can also mean model-view-presenter design pattern.
FOV	Field of view.
FPS	Frames per second.
SLAM	Simultaneous localization and mapping. A technology of mapping an environment while keeping track of an object's location.

1 Introduction

This thesis is an in-depth look in to Microsoft's HoloLens augmented reality headset and its capabilities. For the thesis to be insightful, first there is an explanation of what augmented reality is and in which forms it is used.

The thesis was created by a person who had no prior experience with augmented reality nor HoloLens, so the perspective will be that of a learning journey. The development process of two demo applications for HoloLens will be covered along with best practices and lessons learned from the development processes. The development processes will be followed step-by-step and they include the implementation of each of HoloLens' interaction methods, user experience enhancement with minor visual changes to 3D objects that make for major improvements and the implementation of image recognition.

The thesis was made for Vincit, a Finnish software development company. Vincit offers everything from the development and designing of mobile solutions to Leadership as a Service, with the core business being software development consulting. The thesis was made to serve as an introduction to AR and HoloLens and an information bank for Vincit's personnel and clients interested in AR in general or HoloLens specifically. The purpose of the development of the two demo applications was to get an understanding of HoloLens' capabilities and to create and interactive way of demoing the device to customers.

2 Augmented reality

Augmented reality (AR) is a technology where digital content, often referred to as holograms, is added into the user's view. This technology is close to virtual reality (VR) and they are often addressed together, but whereas virtual reality totally immerses the user by completely replacing the visual world, AR mixes the real world with the digital. [1.]

Augmented reality devices are generally divided into two categories at the moment, there are head mounted displays (HMD) or headsets and phones. Most of the HMDs have see-through lenses, which get holograms projected on them. Another approach to creating an AR device is with video see-through, which uses a camera and a screen to show

the content to the user. Many of the devices also have multiple sensors to make sense of their surroundings and to get the holograms to interact with the real world accurately. [2.]

Phones use the video see-through approach to AR, as nowadays nearly all smartphones have cameras in them. Snapchat was the app to popularize AR applications on smartphones in September of 2015 when the company acquired facial recognition app Looksery [3]. Snapchat was the first app with a user base of millions to implement simple, yet effective and fun image filters, which millions of users nowadays use. [4.] Most of the AR features in today's phone apps are implemented just with the phones' cameras and software. There are some phones, like the iPhone X and Asus Zenfone AR that have extra sensors to boost their AR capabilities. [5,6.]

Augment reality is still relatively young as a technology and it is in the early stages of its lifecycle, but several of the major technology and software companies in the world have been working on their own AR projects for a while now already.

- Apple introduced their ARKit this year at WWDC17 and there are rumours that they are developing AR glasses as well. [7.]
- Facebook have made AR image filters similar to the ones mentioned earlier and they are creating AR development tools by the name of AR studio. [8.]
- Microsoft has an AR headset by the name of HoloLens, which this thesis will cover thoroughly. [9.]
- In 2014 Google released a platform called Tango, which offered boosted AR capabilities to mobile devices. In August of 2017, they followed in Apple's footsteps and announced ARCore SDK and in December 2017 Google discontinued the Tango platform in favour of ARCore. As of December of 2017, the ARCore SDK is still in early preview and available only on few selected devices. [10.]

2.1 History and use cases of AR

The first augmented reality head-mounted display that would show the user simple computer-generated wireframe drawings was created in Harvard in 1968 and in the following decades national agencies, university labs and companies advanced AR technologies in displays and headsets. [11.]

The first commercial use of AR was in 2008, when German agencies created a 3D model of BMW Mini that would appear on a screen when held in front of a computer's camera. The model on the screen could then be controlled by manipulating the physical ad, this campaign was one of the first marketing campaigns that featured interaction with digital object in real-time. [12.]

In the early 2010s, other brands started adopting the same idea, where digital content could be controlled by moving a physical object. At the same time, another use case with the same basic principle became popular. This was "Virtual try-on", where usually a paper printout held in front of a camera to bring out a watch or jewellery on a screen, to see how it would look. Later on, applications that offered a more seamless experience by facial recognition became one of the most successful uses of AR in commercial use so far. [12.]

AR started appearing in museum and tourism use in the 2000s as well, even though they were mostly created in university labs at first. In the past few years it has become more widely available thanks to technological advancements. A use case for museums is to recreate and show a 3D model of an old object or a place to show how it used to look when it was still new and in use. AR also enables the showing of items, scenarios and elements that is just not possible otherwise. [12.]

In 2013 car manufacturers began to use AR in service manuals, Volkswagen developed MARTA (Mobile Augmented Reality Technical Assistance) together with Metaio GmbH to guide the service technicians by showing the next works steps [13.]. A year later Google released the Google Glass prototype for consumers, but the production was stopped a year later, only to come back in the summer 2017 as an enterprise edition [11.]. The HoloLens AR HMD developer kit was released in 2016, with competitor Meta 2 following suite in 2017 [11]. The first entertainment class AR headset by Lenovo, an AR Star Wars experience launched in late 2017 as well. [14.]

Augmented reality powered smartphone applications hit a homerun in the mid-2010s when Snapchat and Pokémon GO gained massive popularity. According to digital marketing agency Omnicore, Snapchat was at over 300 million monthly active users in January 2017 [4], whereas Pokémon GO developer Niantic reported that their game had over 650 million downloads by February 2017 [15].

After the release of Apple's ARKit in the summer of 2017 [7], a number of AR smartphone apps started appearing on the web. Examples of useful and popular apps that utilize ARKit are Place, Chalk, Edmunds and TapMeasure. Place is an Ikea app and with it the users can place 3D models of Ikea's furniture wherever they point at with their phones. The app lets users see if a furniture would fit in a certain space and see how it would look there [16]. With Chalk, users can draw on a video call, with the purpose of the app being to offer the possibility to draw instructions and guide the other user [17]. Edmunds lets the user check whether a certain car model would fit in their garage [18] and Tap-Measure lets the user measure distances and create 3D floor plans quickly. [19]

2.2 Head mounted displays

The competition is tough with AR HMDs, there is a race to the consumer markets as well as competition to establish a major market share in the industry and design sectors. [Image 1] showcases some of the products most popular AR devices and technologies.

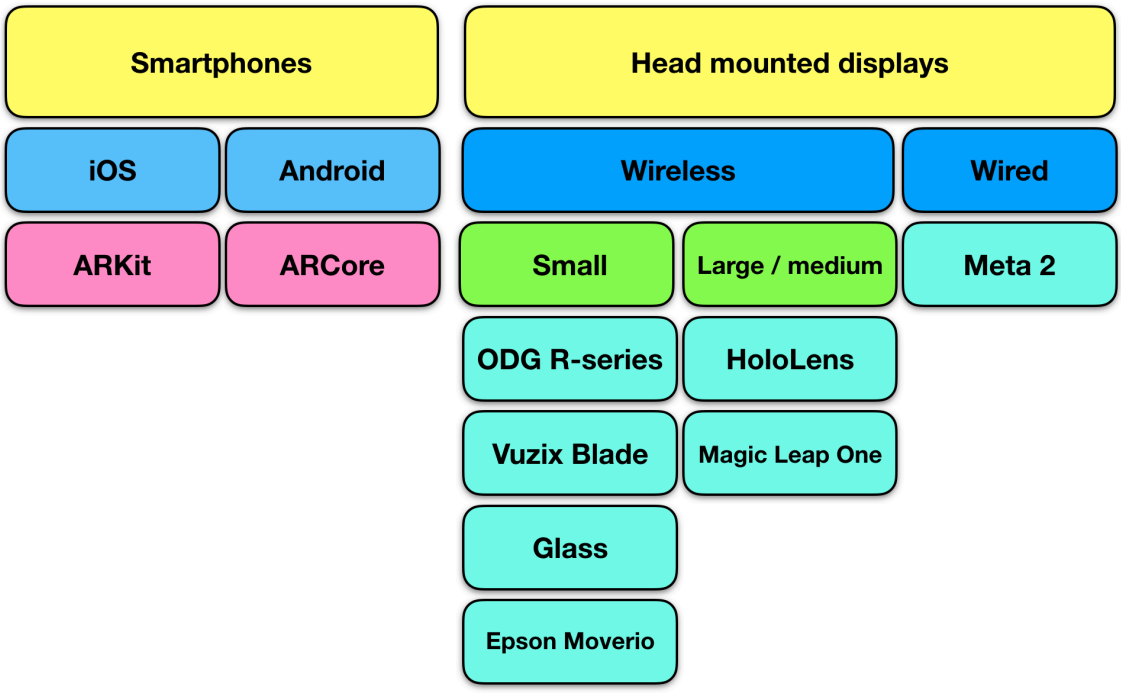

Image 1. Different AR approaches and devices

Meta 2

Seen by many as the main competitor to HoloLens in 2017. Meta's product is a large headset that has to be tethered to a PC. As of January 2018, the device has the widest field of view in the market at 90 degrees (compared to HoloLens' 30 degrees). The device also sports a 2560x1440 resolution display. [20.]

One of the founders of Meta Vision, the company that has made the Meta 2 device, studied neuroscience in university of Columbia. As a result, the company has positions such as Chief Neuroscientist and it puts high emphasis on neuroscience in developing the UX of Meta 2. Meta speaks of their concept as "natural machine", with the idea being that using their device should feel as natural as everyday tasks. [21.]

Meta 2 uses hand gestures/input similarly to HoloLens, but according to reviews, the tracking of the holograms is not on the same level as in HoloLens [22]. Development environment is Unity & C#, with Meta offering their Meta SDK to complement the development. When the Meta 2 launched, limited pre-order quantities were available for $949, with the price being raised to $1495 post launch [20]. While the price of the device itself is less than half of the price of HoloLens, the need of PC which the device tethers to should be considered. Given the specifications of the device and the fact that it has to be tethered to a PC, Meta 2 is well suited for design for design work.

Magic Leap

Magic Leap was a secretive project for several years before the publication of the Magic Leap One. In 2014, the company got $50M in the first round of funding, even though they had only revealed that the project they are working on involves wearable technology, human computing interfaces and ultrasensory perception. No material of their product had been revealed, save for a few teaser marketing videos, until December 18, 2017 [23]. The expectations were high before the product reveal and the company had raised more than $1.9 billion from investors like Google and Alibaba. [24]

Magic Leap One unveiling revealed that the device consists of three pieces, the headwear which contains the displays and some processing power for world detection and machine learning capabilities, a separate processing unit called "Lightpack", which is comparable to a Macbook Pro in performance, and a controller. The device has multiple sensors like HoloLens and apparently the field of view is slightly larger than in HoloLens.

An SDK is supposedly coming in early 2018, with the first devices shipping later in the year. [25.]

ODG R glasses

Sunglass shaped smart glasses which run on Android Nougat based OS, the upcoming ODG R9 model has a FOV of 50 degrees, 1080p resolution, six degrees of freedom tracking, utilizes SLAM technology and a Qualcomm's Snapdragon 835 chip as a processing unit and good overall components as seen in [table 1]. The device can be reserved as of December 2017 with a price of $2199 for international orders. [26.]

Upgradeable parts like low-light vision and enhanced environmental scanning are planned for the device as well. Interaction with the device is handled with buttons and controls on the glasses, or optionally with a smartphone app. The ODG R9 might get gesture input cameras after launch. [27.]

The small form factor combined with good technical components presumably make it good for hobbyist and industrial use. ODG has made a version suited for hazardous conditions of their previous smart glasses, the R7, so they have experience designing their product for industrial environments. [28.]

Table 1. Technical specifications of ODG R9's components. [26.]

Components	**Technical details**
CPU	Qualcomm Snapdragon 835 @ 2.45 GHz
RAM	6 GB
Storage	128 GB
Battery	1400mAh (2x 700mAh)

Vuzix

Vuzix Blade smart glasses are much like the ODG R9, shaped just like regular sunglasses. The developer kit of the glasses can be reserved as of January 2018 for a deposit of €570, with the final price being €2292. The glasses run on Android OS, on a

quad core ARM chip, with detailed technical specifications still being kept secret. Revealed control methods for the device are voice control, touch pad with gestures on the arm of the spectacles, head motion tracking, haptic feedback and a companion app for Android and iOS. As the product resembles ODG's glasses a great deal in all aspects, the use cases for the Vuzix Blade are similar. Whether the Vuzix Blade and ODG R9 smart glasses are good enough and useful enough for everyday activities remains to be seen. [29.]

Google Glass

Google is bringing back the Google Glass, this time as an enterprise edition, aimed specifically at businesses. The device itself is a small wearable computer with a transparent display that can be clipped onto glasses and safety goggles for example. Based on the Glass website, it seems like the main focus with the device is to make workers more effective in areas such as manufacturing and logistics. As of January 2018, the Glass enterprise edition is available exclusively through companies acknowledged Glass Partners. [30, 31.]

Epson Moverio

The latest version of the smart glasses is the BT-350, a consumer/hobbyist device with 720p resolution and a 23-degree diagonal viewing angle, able to project an 80" virtual screen 5m away. Uses Android 5.1 and has a battery life of approximately 6 hours. Epson also manufactures industrial models for construction etc. the BT-350 glasses are available from $1399. [32.]

DreamWorld DreamGlass

Not much has been revealed of this headset yet. The device looks like a HoloLens/Meta 2 type headset with 100-degree FOV with HD resolution displays and support for hand gestures, head tracking and face detection. [33]

Varjo Alpha prototype headset

New Finnish start-up that promises 70mp super high resolution in an "Extended Reality" headset, originally built on top of the Oculus Rift VR-headset. Apparently, the headset

uses eye tracking technology to display a high-resolution area wherever the user is focusing on. The device seems like more of a VR technology at this state. [34]

3 Microsoft HoloLens

One of the most advanced and well-known augmented reality headsets is Microsoft's HoloLens [see image 2], released in 2016. The headset is completely wireless, with battery life of about 2–3 hours in active use. The availability of the device was limited until November of 2017, when Microsoft expanded HoloLens' availability to more countries in Europe, including Finland. As with most of the currently available AR headsets, the HoloLens that is on sale right now is a development edition, with a price of 3 349€. [9.]

Image 2. HoloLens augmented reality glasses [9].

3.1 Technical specifications

Microsoft has been secretive with detailed technical specifications of HoloLens. It has revealed that the device has two HD displays with a Field of View of 30 degrees but has not revealed the actual resolution of the displays. Performance wise, it is on the level of other higher-end mobile devices [table 2]. On top of a mobile CPU and GPU, HoloLens has what Microsoft calls a Holographic Processing Unit. [35.]

Table 2. Technical specifications of HoloLens' components [35].

Components	**Technical details**
CPU	Intel Atom x5-Z8100 @ 1,04 GHz
GPU/HPU	Microsoft Holographic
RAM	2 GB
Storage	64 GB
Battery	16,500 mWh

The device has cameras and sensors for rotational and positional tracking, depth measurement and hand gesture recognition as well as microphones for voice commands. The device contains the following sensors:

- an IMU
- four environment sensing cameras
- a depth camera
- a photo/video camera
- four microphones
- an ambient light sensor. [35.]

Hologram distance guidelines

The draw distance in HoloLens is from 0.85m to basically unlimited, but the recommended hologram range is from 1.25m to 5m, with 2m being the sweet-spot. Holograms closer than 2m or farther than 5m may cause eye strain. [36.]

Lighting restrictions & Outdoors functionality

HoloLens seems to work best in "neutrally" lighted or slightly dim areas, as the device has trouble mapping the surroundings in the dark, and bright lights cause poor visibility. With maximum brightness, the HoloLens works adequately outdoors in shadowed areas or on a cloudy day, but the visibility is bad in direct sunlight. Testing carried out at Vincit

[37.], revealed that the device picked up voice commands well despite minor traffic noises and wind.

3.2 Interaction

HoloLens has the three following ways of handling user interaction.

Gaze

The dot that is often displayed at the center of the user's line of sight and the primary form of targeting on HoloLens.

- recommended to be circular and not directional
- can be hidden
- A directional arrow/pointer can be added next to the cursor/target to guide the user to look somewhere. [38.]

Gestures

At the moment, there are no built-in features within unity or HoloLens SDK to record gestures. The following are the default gestures:

- tap: select (can also be called with the voice command: "Select")
- hold: open a menu
- manipulation/navigation (hold + drag): scroll, zoom, rotate
- bloom: go back to Start Menu (voice command: "Hey Cortana, Go Home")

Hand recognition: HoloLens detects both hands and sees when they are in **ready state** or **pressed state** and ignores other poses. The position (without orientation) can be accessed for each hand that HoloLens detects, and its pressed state. As the hand nears the edge of the gesture frame, a direction vector is also provided, which can be show to the user, so they know how to move their hand to get it back where HoloLens can see it. [39.]

Voice input

Developers can add their own voice commands simply by adding strings to a keyword collection. The voice commands work with gaze and they can be target specific or global. They are good for traversing complex interfaces as it lets the user cut through nested menus with a single command. HoloLens uses the same speech recognition engine that supports all other Universal Windows Apps.

It is best to use commands that are concise and have multiple syllables, as one syllable commands are harder to recognise by the device. An example of a good voice command is “Play video”, and a bad one is “Play the currently selected video”.

Tips:

- Use simple vocabulary
- Be consistent
- Don’t re-use default keywords such as “Select”
- avoid similar/rhyming commands

Default keywords:

- “Select”: Same as a tap gesture or clicker click,
- Hey Cortana,
  - Go Home,
  - Launch <app>,
  - take a picture,
  - mute [40.]

3.3 How the device works

Spatial Coordinate System

HoloLens uses Cartesian coordinate system (X, Y, Z) with metric values. The digital content that is displayed to the user can be world-locked (where the rendered object stays put when the user moves), or body-locked (where it follows the user). [41.]

It is advised that most content is displayed as world-locked by default, but in scenarios where HoloLens' sensors cannot gather enough data about its environment, it is good practice to show body-locked content that would guide the user to check if something is blocking the sensors or to turn on lights, for example. [41.]

Spatial Anchors

Spatial anchors are points that have their own coordinate systems that adjust themselves relative to other spatial anchors or points in the coordinate system to make sure that anchored holograms stay in place. With saving and loading of spatial anchors to and from the device's storage, an application can make sense of real world locations across different sessions.

Rendering a hologram in an anchor's coordinate system achieves the best results in the positioning of a hologram, with the downside of small adjustments to the hologram's position as time passes, as the system re-adjusts the hologram back into place relative to the real world. It should be noted that, once an anchor is placed, it cannot be moved. [42.]

Spatial mapping

Spatial mapping is the method HoloLens uses to map / make sense of the area it is in. The device does it by scanning in a 70-degree cone region between 0.8 and 3.1m away from it and drawing a mesh on what it sees [image 3]. Spatial mapping can be used to detect walls, ceiling, floor, tables and the gathered information can then be used to spawn objects in correct places, for example a poster on a wall and a cup on a table.

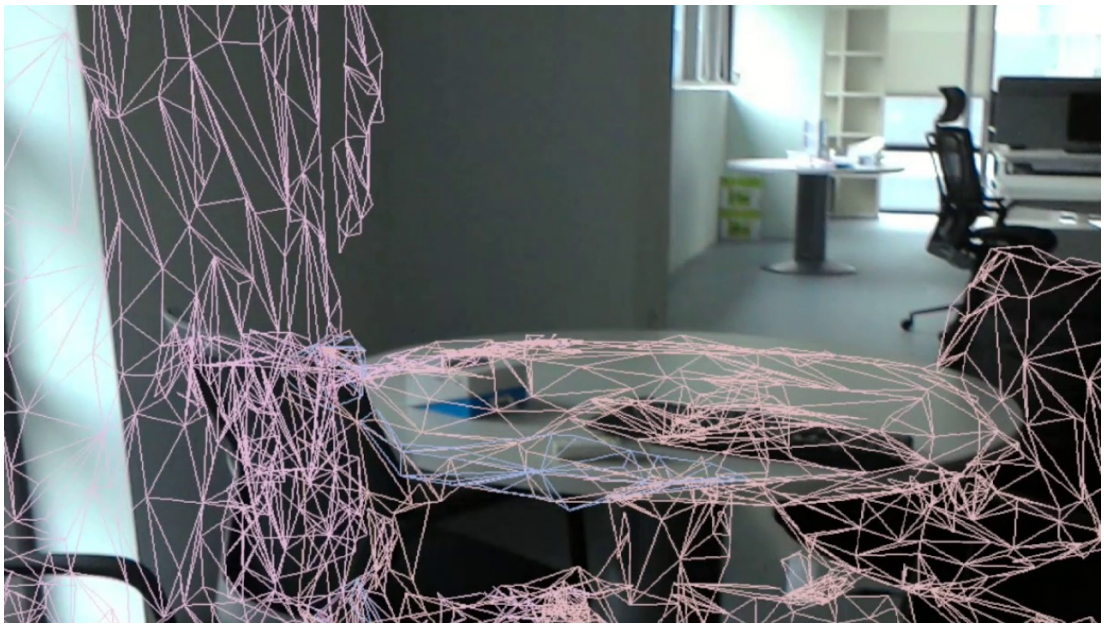

Image 3. Screenshot displaying the spatial mapping mesh. [43.]

Main purposes

- Help holograms navigate
- physics
- occlusion
- placement
- visualization

Types of room scans

- One-time

Best for static/semi-static environments where holograms do not need to react to changes in the environment. Uses more disk space from the device but is more efficient performance wise.

- Continuous

Best suited for dynamic environments where holograms need to react to the environment. Continuous scanning has higher CPU impact and therefore and affects performance and heats up the device more.

Applications can be made to rely on mapping data that has already been collected, or they can implement their own scanning experience. If the application has its own scanning section, the user should be guided in it by instructing the user to walk and look around. The storing of the room scanning data is always done by the system, so the application does not need to do it. [43.]

The mapping mesh's accuracy value can be changed up to 2000 triangles per cubic meter, with the default value being 500. [44.]

Spaces

HoloLens saves hologram and spatial mapping data to areas called "spaces", which it links to Wi-Fi networks it detects. The device needs to have Wi-Fi turned on to get the previously mapped space -data, but it does not need to be connected to a network. When setting up a place, the more the user looks around, the more accurate the device's mapping data will be. Spaces can be shared with other HoloLens devices. [45.]

Holograms & comfort

There are steps that human eyes have to take for them to see objects clearly. Two of them are important when it comes to AR and VR headsets. The first one, accommodation, is when an eye adjusts it's focus on a distance to see a clear image. The second one, vergence, is the eyes' focus relative to each other. The eyes must converge to an object's distance not to see double images. Naturally eye accommodation and vergence are linked, but with HMDs, the user's eyes will accommodate to the distance of the display while the vergence will change according to the distance of the object of interest. This breaks the link between accommodation and vergence and may cause eye strain and discomfort. [36.]

HoloLens' displays are approximately 2m optical distance away from the user's eyes, which is why the user should be about 2m away from a hologram for the image in the device to stay clear. In scenes where there is a great deal of content at different depths, the areas of interest should be placed as close to 2m as possible to reduce discomfort. In cases where it is not possible to place the content at 2m, it is best to try to keep the

target objects at a fixed depth. Because of the vergence-accommodation conflict, situations where the user has to focus back and forth at different distances is more fatiguing to the eyes than looking at stationary targets. [36.]

As mentioned earlier, the optimal distance for holograms to be placed at is between 1.25m and 5m away from the user. It is considered good practice to set the clipping plane to 0.85m and start fading out holograms closer than 1m as the odds of discomfort due to vergence-accommodation conflict multiplies with decreasing view distance. With applications where holograms need to be displayed close to the user, it is recommended to have a “depth budget”. Depth budget means limit to the time that holograms are close to the user, and in cases where the limit is exceeded, careful user testing should take place to ensure that the experience is not too discomforting. [36, 46.]

Keeping the application’s FPS at a steady 60 is crucial for a smooth and comfortable experience when there are moving objects in a scene. As an application’s framerate drops, judder appears in moving objects in the form of uneven motion and double images. In addition to high FPS, framerate consistency is important as well. Consistent FPS of 30 looks more stable than FPS fluctuating from 60 to 30 and back. [46.]

FPS higher than 60 would be even better, but unfortunately HoloLens’ displays are limited to showing content at what looks like 60 FPS to the user. For comparison, the first high-grade consumer VR headsets Oculus Rift and HTC Vive have 90Hz displays, meaning that they are capable of showing content at 90 FPS. [47.]

3.4 Functionality

Hologram sharing

Hologram sharing is possible and with MixedRealityToolkit it’s done following these steps:

- One pc sets up the sharing service and deploys first. The anchor point will then be uploaded, once the point is uploaded, a hologram can be placed on it.

- Once a hologram has been placed, other devices can launch the app and once their device has downloaded the hologram and anchor point locations, the hologram should appear at the point where the first used added it.
- Walking to where the original hologram was placed and gathering environment clues can help with problems with the hologram location.

Real-time sharing works best as one-to-one or with a small group (less than seven people). According to Microsoft, the complexity increases exponentially when going from six to seven people. [48.]

Live streaming

Live streaming from the device is possible, though it drops the fps of HoloLens to 30, making the experience less smooth. Microsoft has provided two ways for it so far, one is through the browser in Device portal with the Mixed Reality Capture-section's Live preview, but during tests carried out at Vincit, there was a lag of 4–6 seconds. The other way of streaming is through a Microsoft HoloLens desktop app, available in the Microsoft Store in some regions (not in Finland at the moment of testing, but accessible by changing the PC's region to United States/Canada). With the desktop app, the lag was noticeably shorter at around 0.5-1.5s and there was an option to decide on either Higher quality or faster stream quality. Even with the High-Quality stream, the resolution as well as the fps are far from great, but it was still adequate. Streaming to iOS can be done with an app called HoloStream. [37, 49.]

Bringing 3D models to HoloLens

Users can import their own .fbx or .obj 3D models from 3DS Max, Maya or other 3D modelling software to Unity and then either place them in a game scene in the editor or make a 3D asset out of them and export them to OneDrive to use them with HoloSketch or other applications that can show 3D assets. A thing to keep in mind when importing 3D models from other software to Unity, is the model's rotation in Unity, as some software use different coordinate systems. [50]

Through the Device portal, it is also possible to download the spatial mapping mesh of the environment that HoloLens has created to a computer and import it to Unity for example. [37.]

3.5 Use cases

Japan Airlines

Japan Airlines has created an app for HoloLens which lets their trainee engineers see and interact with working holograms of jet engines. This lets them train on complex machinery that is hard to access. [51.]

Nasa

Together with Microsoft, NASA has developed OnSight, a software that they use with the Curiosity Mars rover project to help the scientists visualize Mars. NASA and Microsoft have also created an exhibition called Destination: Mars, where the visitors of Kennedy Space Center Visitor Complex get to virtually visit sites on Mars, reconstructed of imagery from the Curiosity Mars Rover. [52.]

Volvo

Volvo has made a showroom app for their cars, allowing the HoloLens users to see a life-size 3D model of a car, with features like trimming away the body of the car to see the chassis at work. They have also visualized the how different sensors work in their cars etc. A dealership showcasing this app for customers, for example to demonstrate different customization options of a car could drive the sales quite nicely. [53.]

Fragments

One of the most convincing showcases of HoloLens' capabilities is a game called Fragments, developed by Asobo Studio, in co-operation with Microsoft. The game is about crime scene investigation, and what makes it so compelling is the way it interacts with the environment. [54.]

When the game is first started in a new area, it asks the user to walk and look around to map the play area with spatial mapping to gather data about the environment. After having gathered enough data, the game makes player's room a crime scene, fitting it specially for that space. If the room is small, the game leaves out some of the less important

objects and prioritizes on the critical elements. The game does a good job of fitting objects in places where the user would expect to see them, a coffee cup and a paper on a table for example. This was achieved by utilizing spatial understanding code found in the MixedRealityToolkit. [54, 55.]

Image 4. A game character utilizing HoloLens' spatial mapping data to recognize surfaces categorized as seats sit on a sofa [54].

What makes for most of the immersion though is the characters. The characters are life size and look realistic enough not to distract the user. The characters' behaviour is well executed, they look the player in the eyes when they talk, navigate through the play area smoothly and use actual chairs to sit on [image 4]. [54, 56.]

3.6 Development

Most of the 3D applications for HoloLens have been made with Unity and with C# programming language, even though it is possible to use other engines as well. As HoloLens uses Universal Windows Platform (UWP) and the development is done in Unity, there are some compatibility issues with frameworks and libraries for .NET environments. For example, the popular .NET JSON-parsing library, Newtonsoft's Json.NET is not compatible with HoloLens projects. [57.]

HoloLens also supports normal 2D UWP applications, but the interaction with such applications is limited compared to 3D, as 2D applications cannot utilize the sensors to the same extent as 3D apps. 2D HoloLens applications can however utilize gaze, gestures and voice commands. [58.]

When a 3D app on HoloLens is started, the device hides all of the operating systems UI and elements, the effect is similar to opening a full-screen application on a PC. Whereas only one 3D application can run at a time, it is possible have several 2D app windows visible at the same time. While many 2D apps can visible at the same time, only one of them can be active at a time. This means that 2D apps have to be focused on / actively selected for them to work, when a second 2D app is selected after using the first one, the first one will go to a paused state. [59.]

The manipulation of 2D apps also can be done only with HoloLens' native manipulation methods, for example, when a user wishes to move the application, they first have to select the moving option from the applications top panel, instead of using a voice commands to move the object. [56.]

Image 5. Marketing picture of HoloLens' holograms, the image can be misleading. [39.]

HoloLens has a pre-installed application called Holograms, which allows placement of singular holograms in a space, like the ones that can be seen in [Image 5]. As of yet however, the Holograms application is the only exception where this can be done. Developers are unable to create singular holograms that could run or be displayed with other such holograms in the operating systems main view, as there are no public APIs available for it yet. [59.]

The scenario in [Image 5] would have to be inside a single 3D application. This means that the application would have to be running a video player, be getting the weather data by a HTTP request, as well as have the functionality for the to-do list and the recipes.

A webcam can be used in the Unity editor's play mode to mimic HoloLens' camera to hasten development, but it is good practice to test on the actual device too, as the experience might be different. [37.]

Requirements / setting up the development environment

HoloLens development requires a Windows operating system and Windows 10 SDK. The recommended version of the operating system to use is Windows 10, but the SDK works in Windows 8.1, 8, 7 and Windows Server 2012 as well as Windows Server 2008 R2. [57.]

Microsoft has released a set of tools called MixedRealityToolkit (formerly HoloToolkit) to speed up the development process, with button elements, code for dragging holograms and other helpful files. Another helpful tool for HoloLens development is the HoloLens emulator, as it allows for development and testing of holographic applications, without the need for the expensive device. Although the development can be done with just the emulator, it is considered good practice to test applications regularly on the actual device as well. [60.]

4 Research and development process

The goal for the practice portion of the thesis was to research what is possible to do with HoloLens and learn about the device's capabilities and restrictions, as well as to develop a demo application for HoloLens. The purpose of the demo application on top of putting the knowledge to practice was to get an application that could be showcased to customers.

4.1 Setting up the environment

The development was done on a MacBook Pro, but as a Windows operating system is mandatory for HoloLens development, it was installed on top of Parallels Desktop 11. As the demo application to be developed was supposed to be a 3D application, Unity 5.6 was used for the development of the app. As Vuforia SDK was not yet integrated in to Unity 5.6, it was downloaded and added to the project separately, around mid-way through the development process of the first application along with MixedRealityToolkit. Visual Studio 2017 was used for coding and deployment to the device, so in the end, the software stack for the final year project was as pictured in [image 6].

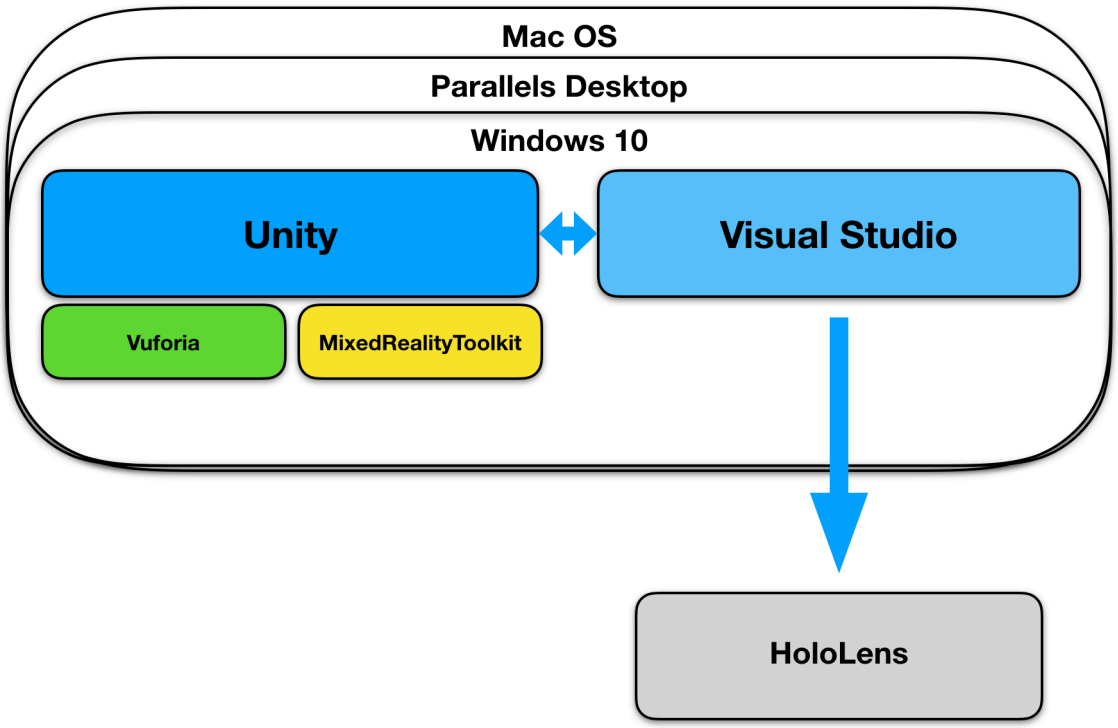

Image 6. The software stack used for the final year project.

Despite the whole development environment being on top of an emulated Windows OS, there were very little problems with it. There was a minor quality-of-life problem with the performance in the Unity editor, as moving the camera around was not smooth at all, but that did not slow down the development much. One thing to note is that the emulator was not needed for the development, as an actual device was available for the whole length of the process. The hardware requirements of the emulator on top of the emulated Windows OS could have made for significantly slower testing of the applications.

4.2 Getting started with the development

Microsoft offers plenty of resources to get started with HoloLens development in their Mixed Reality Academy. They have tutorials for implementing each of the interaction methods for the device. The tutorials usually contain videos with step-by-step implementation with an instructor in Unity editor and a text guide that covers the same things as the video. MixedRealityToolkit is used in most of the videos, but as the toolkit receives updates regularly, some of the videos have instructions that are not up to date.

The development process started from zero, as HoloLens had limited availability globally until November of 2017, the device being originally released in 2016, and with the price being over 3000€. Due to the lack of experience with the device, the first 10 days of the development process were spent on reading about how HoloLens works, guidelines about hologram comfort and overall getting familiar with the device. During that time, all of the available tutorials in Mixed Reality Academy that covered HoloLens development were completed to get an understanding of each of the interaction methods. As of January 2018, there is also a tutorial about the motion controllers for the VR-like Windows Mixed reality headsets.

After the first 10 days, weekly meetings started to take place to go over the progress of the project. In the first meeting, it was decided that the first demo application should be something relatively simple. A 3D application was to be made, that would get data from some open API and display it to the user, so it was decided that the project would be a weather app.

4.3 Weather window demo application

From there, guidelines were studied on how text should be displayed to the user in a holographic app. The first step in the Unity editor was creating a plain 3D cube and attaching a 3D text element to it, with a default value of “Waiting for connection…” that would be replaced by the data received by a HTTP get request.

Unity can emulate threading with coroutines, so that was the way to implement the network operation as seen in [image 7].

```
public void getData(string city)
{
    StartCoroutine(getJson(city));
}

IEnumerator getJson (string cityName) {
    Debug.Log("GetJson called");
    string url = "http://api.openweathermap.org/data/2.5/find?q="
    + cityName + "&units=metric&appid=43774a7759004745b94f432c5fd311c9";
    www = new WWW(url);

    yield return www;
    if (www.error == null)
    {
        ProcessJson(www.text);
    }
    else
    {
        Debug.Log("ERROR: " + www.error);
    }
}
```

Image 7. Running the HTTP get coroutine

After learning how to use the coroutines, getting the data from the openweathermap’s API was simple enough, but then there were problems with the parsing.

Unity has JsonUtility for working with JSON data, but it cannot handle nested JSON objects and its main purpose is serialization. Several popular .NET and C# JSON libraries were looked in to and the implementation a few of them was tried. Some seemed to work well and had no hiccups when testing the app in the Unity editor but would give

errors once trying to deploy the app to HoloLens. In the end, a working JSON library called SimpleJson was found in the Unify Community Wiki. [Appendix 1]

```
//parse JSON, update data to the window & display correct icon.
private void ProcessJson(string jsonString)
{
    var N = JSON.Parse(jsonString);
    string message = N["message"];
    city = "City: " + N["list"][0]["name"];
    temp = "Temp: " + N["list"][0]["main"]["temp"] + "°C";
    min_temp = "Minimum temperature: " + N["list"][0]["main"]["temp_min"] + "°C";
    humidity = "Humidity: " + N["list"][0]["main"]["humidity"] + "%";
    weatherType = N["list"][0]["weather"][0]["main"];
    description = N["list"][0]["weather"][0]["description"];
    textMesh.text = city + "\n" + temp + "\n" + min_temp + "\n" + humidity;
}
```

Image 8. code snippet for parsing the nested JSON objects and displaying the selected city, it's current and minimum temperature as well as the humidity on the window's text element.

With the parsing problem out of the way [image 8], it was time for another weekly progress meeting. So far, the application was just a rough prototype, displaying just a plain box with London's current weather data. As the window looked very dull as it was [image 9], goals were made to make it more engaging and interactive.

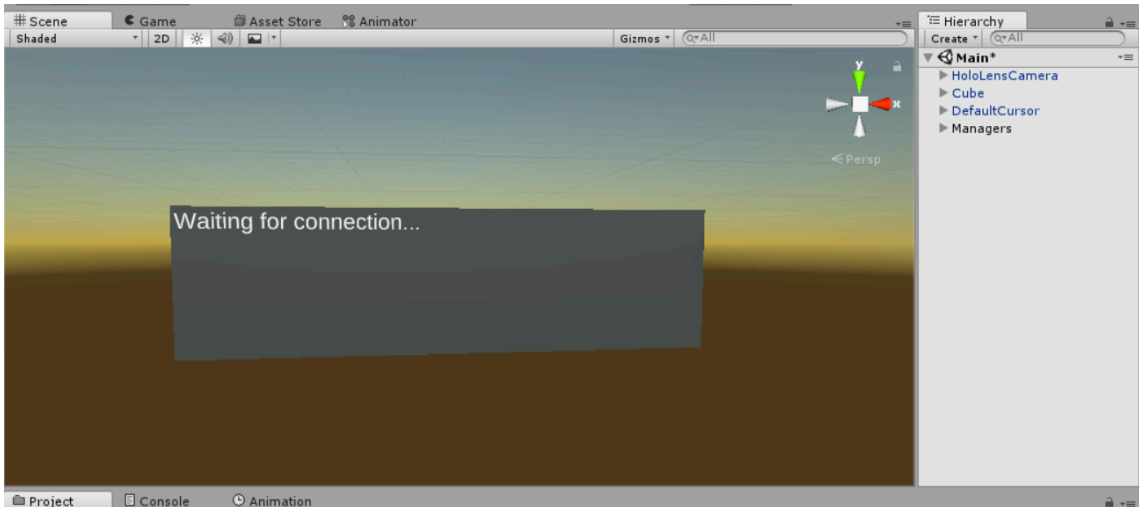

Image 9. The application in Unity editor in early stages of development

Based on the meeting, the next step was to make window more pleasing to the eyes by giving it a red slightly transparent glass-like material from HoloToolkit. After the material

change, buttons were added to the top of the window to allow the user to get data from a few other cities. The buttons did not make for a great user experience as they were, so looking for ways to improve them was the next task.

The process continued with the creation of an animation indicating the buttons' state. When the user's gaze hovered moved to a button, its colour would slightly change, and it would move backwards about the same amount as the button's depth. After the back and forth animation, it was made so that when the user selects a city by gazing at it and air tapping, the selected button would get a highlighted colour to make it stand out from the other buttons. It took surprisingly long to get the buttons' animations just right, after having finished them, it was time for another weekly meeting.

The mentors were pleased with the progress, as the window looked significantly better at this point, it fetched data from an API, allowed the user to switch between a few cities with buttons on top of the window and the window had a better-looking material now. As the application was being developed by one just person, the weekly meetings were a good chance to get feedback from other people who had not spent as much time staring the weather window. Some good feedback that was given in the meeting was to make the window smaller, as HoloLens' FOV is quite limited, it was easy to get the screens to clip the hologram and not be able to see the whole window.

After having made the window shorter in height, the buttons' user experience was still not satisfactory, there should've been more feedback from air tapping a button than just the highlighted colour. Adding a clicking sound to the event improved the experience considerably. It was little things like this that made a big difference in making the holograms seem more realistic.

So far, the text that was displaying the data on the box was Unity's default 3D text element, and the text's resolution was low, despite a few fixing attempts. This problem was fixed by installing an asset called TextMesh Pro, from Unity's asset store. The text element created by TextMesh Pro had no jagged edges and made the application look more polished.

Implementing voice commands was trouble-free for the most part, thanks to MixedRealityToolkit's speech scripts. The first voice commands that were added were the same cities' names that were in the buttons, so instead of the user clicking on the button with

“Austin” on it, he could just say the word. To give feedback to the user about which keywords were recognized by the device, a text row displaying the recognized words was added under the weather data.

At first a mistake was made by adding the speech recognition scripts on the window game object, making it so that the application was listening for and registering keywords only when the user’s gaze was on the window. It was hard to realize this, as it was hard to know whether the application was working faultily, or if the word had just been pronounced wrong, so that the device did not recognize it. Moving the speech recognition scripts to a manager game object, separate from the window, fixed the problem.

Hand dragging was the next thing to be implemented. It was made so that when the user’s gaze was on the window, he could air tap and keep his thumb and index together to “grab a hold” of the hologram. The hologram could then be move in each direction, including in the Z-axis, as long as the fingers were pressed together. Also, a helper voice command keyword “Show window” was added, to move the window from wherever it was to in front of the user’s head and have it face the user.

During testing it came apparent that it is quite easy for the user to move the hand that was dragging the hologram out of the area HoloLens’ cameras and sensors cover. In cases where this happens the hologram stops moving and if the user releases his grip of the hologram outside the covered area, the device does not register the action of letting go and the user might get confused. After some testing an icon to display when the user had a hold of the window was added next to the cursor.

The last feature to be added to the project was either text or image recognition. After a couple of ways to implement text recognition were looked in to, Vuforia SDK was chosen as the solution to be used. Vuforia seemed like a good solution due to its capabilities in both text and image recognition. After having configured Vuforia in Unity, the word “Berlin” was printed on an A4 with large letters. It was made so that when the application recognized a word, it would mark and track the word with clearly visible frames, and it worked great in the Unity editor’s play mode with the laptops webcam. When testing out the feature with the actual device, the experience was much worse. The device had a hard time recognizing the word and when it did, the frame was jittery and disorienting.

As Vuforia's text recognition did not work well, Image recognition was up next. With the image recognition, the images have to be uploaded to Vuforia's server where they analyse the images features and give it a rank from one to five, depending on how easily it can be recognized. After uploading the images that are going to be used in the application, a database file of the image collection has to be downloaded and added to the Unity project.

For this project, an image of the Brandenburg gate was added to the database, and it got a rating of 5, so it had good recognisability. The functionality for recognizing the image was set to the same as when "Berlin" keyword was recognized, or the corresponding button pressed. Once again for feedback, a sound was added, to be played whenever the image was recognized.

Now all the functionality had been implemented and as final polishing, animated icons were added to reflect different weather conditions. An icon of a sun was added, with a script attached to it to make it spin, for clear weather, a cloud moving back and forth for a cloudy day and both of the icons were visible on a partly cloudy weather [image 10]. Once again, this addition was nothing ground breaking, but it added to the engagement and made the window a bit more interesting.

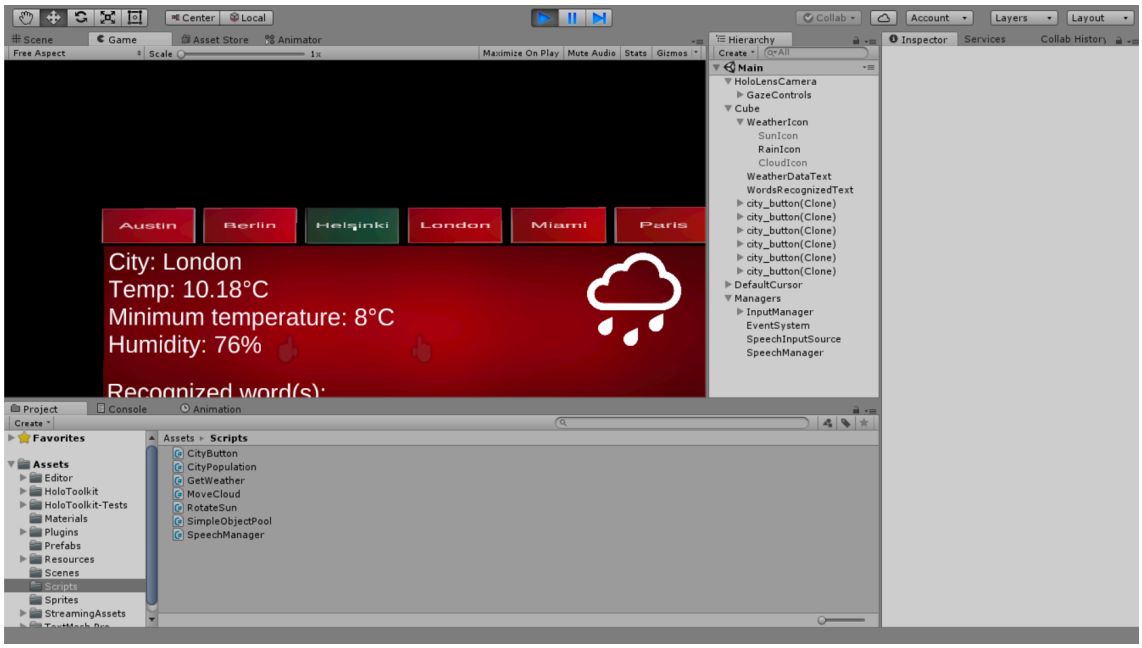

Image 10. Weather window in Unity editor's play mode in a late stage of development. Demonstrating the look of the window and city button's gaze hover highlight.

At this point, the demo application was ready to be demoed in-house. The presentation went well, the different features of the application were tested step-by-step, with the last step being image recognition. When the presentation ended with recognition of the Brandenburg picture, plans were made to demo the application to a customer that was interested in an AR project with HoloLens.

Shortly after, a meeting with the customer took place and the weather window application was demoed to them. The customer liked the idea and the utility that a device such as HoloLens offers, but using HoloLens in particular was not an option for them, as the device did not fit under a construction helmet. A smaller, yet capable device such as the ODG R9 could have been a good fit, if only it had been available for purchase.

While the development of the weather window app was ongoing, Microsoft Build Tour was held in Helsinki for the first time and a few of Vincit's employees attended the first day of the conference. There were high hopes of getting insight into more advanced topics of HoloLens development and to learn about the then-mysterious Windows Mixed Reality headsets. Unfortunately, the only way HoloLens was covered was a short demo, where a hologram appeared as the device recognized a Vuforia image target. Acer's Windows Mixed Reality headset was shown on stage just as quickly, without a chance for visitors to get any hands-on experience.

To sum up the development process of the weather window project, when starting development for a cutting-edge device such as HoloLens, with no previous experience with the device or augmented reality, the progress comes in small steps. After the initial planning, the process on implementing each feature was as follows:

- Decide what to add next
- learn/read how to do it
- implement it
- test it
- polish it.

As with many software development projects, the core functionality was added first, then the visuals were improved, as changing the material made a big difference in how the hologram looked, some interactivity was added and the image recognition feature which

had a big impact on testers as well and lastly some minor quality-of-life tweaks to enhance the overall experience.

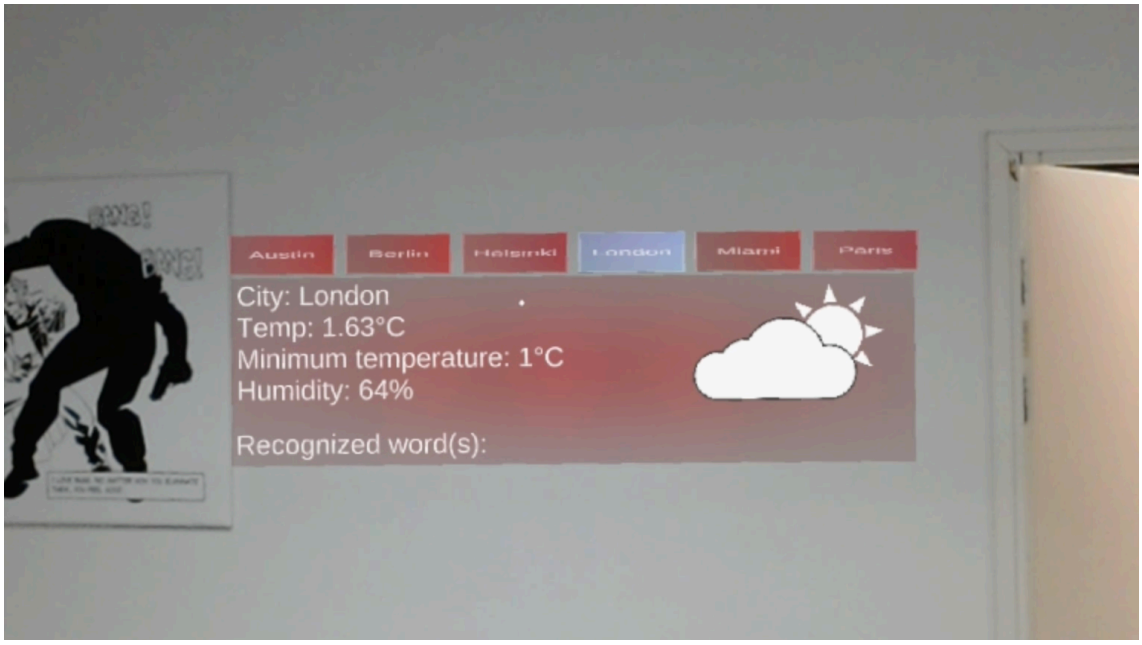

Image 11. A screenshot of the finished version of the weather window demo application. The hologram quality does not represent what is seen through the device's lenses. The resolution is better, colours are more vibrant and the overall visibility of the hologram is better in reality.

The weather window demo app was now finished [image 11] and now that some experience had been gained from HoloLens development, a planning of another demo application started. The plan was to make something where the core idea itself was interesting and fun or useful and not just rely on the holograms to impress the user. There were several videos available on the internet already, where a guidance or a navigation application had been made to show the way to a certain place in an office, so the idea had to be something else.

Before development of any proper new ideas started, some 2D prototypes were developed quickly, just to see what the process was like and how a 2D application differs from a 3D application. There were no surprises with the development, the process went the same as any other UWP app and instead of using Unity with Visual Studio, all of the work could be done in Visual Studio alone. Even though Unity and Vuforia have limited free personal licenses for hobby use, both of the platforms have licence costs for business use, so the upside of an UWP app would be the lack of these costs.

During a planning meeting of the next demo application, someone came up with the idea of counting cards. This idea filled all the pre-requisites that had been decided on, it was an exciting out-of-the-box idea and so it became the next project.

4.4 Blackjack card counter demo application

There are several common ways of counting cards, but the basic idea is to assign values to each card and to keep track of the running count. For example, high cards are given a value of -1 and low cards are given a value of +1, with the sum of all the card values being the running count. The running count being high means that the ratio of high and low cards is such that there are more high cards in the shoe (where the dealer deals the cards from), which is favourable for the players.

A second developer joined the planning and development process of the blackjack app. This developer had no previous experience with HoloLens and the timeslot he had for this project was just two weeks, so the plan was to develop a minimum viable product type of prototype of the application first and possibly continue from there.

This second project was made in Unity as well, and as Vuforia's image recognition feature had left a good impression, it was chosen as the way to get the device to read the cards. At first, Bitbucket was considered for version control, but Unity's built in version control system was chosen for due to its ease of use and functionality. First a regular looking deck of cards was selected, as to make the application work in as many scenarios as possible with just one deck. The cards were then scanned one by one and added to the applications Vuforia image database.

The cards' recognisability rating ranged from one to five, with High cards that had many distinct features getting the high ratings, and low cards getting low ratings due to few features in the card's image [image 12].

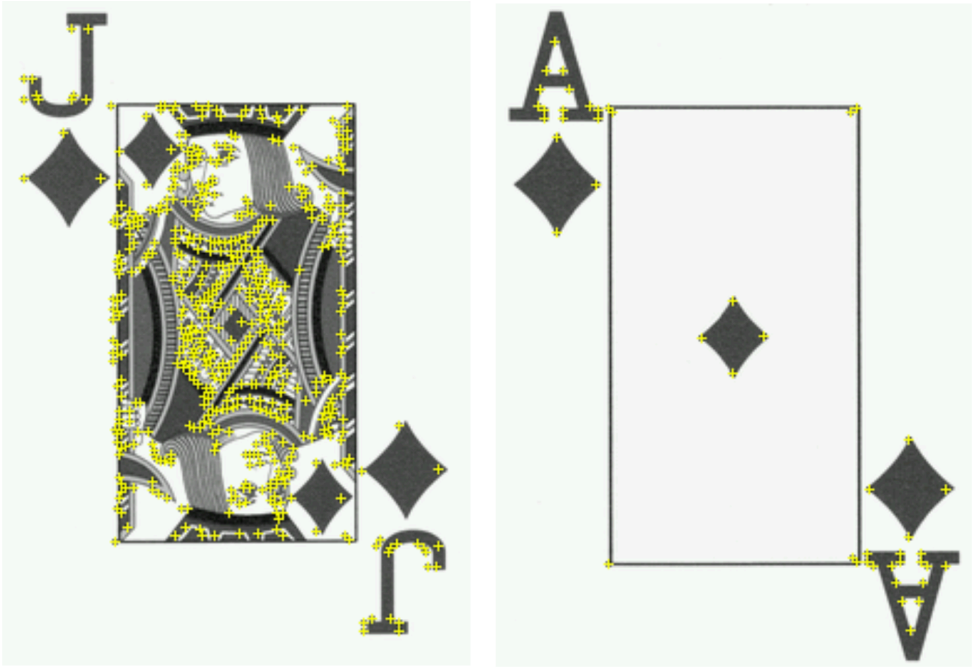

Image 12. A demonstration of Vuforia's image recognition, with the recognizable features of the images visible with the yellow signs. The Jack having a rating of 5, with the Ace having a rating of 1.

To start off, the app was made to take into account only one player's (the HoloLens user's) cards, not caring about possible other players in the table. The app would instruct the player in two ways, the first and simpler instruction to implement was to show the probability of busting (getting a total card value of over 21, which makes the player lose that round). For example, if the player gets cards 9 and 7, totalling 16, the player would see the text "Chance to bust on a hit: 62%".

The second instruction was to show the recommended strategy [Appendix 2] in each scenario, with four options: Hit, split, stand and double. The logic was so, that the user would first scan their two own cards and then the dealers card. After three cards had been recognized, the suggested strategy would be shown [image 13].

```
public void AddCard(string trackableName)
{
    if (GetPlayerCards().Count < 1)
    {
        AddPlayerCard(trackableName);
        ChangeText.Instance.SetCardText("Player card 1: " + trackableName);
    } else if (GetPlayerCards().Count == 1)
    {
        AddPlayerCard(trackableName);
        ChangeText.Instance.SetCardText(
            "Player card 1: " + playerCards[0].card
            + "\nPlayer card 2: " + trackableName);
    }
    else
    {
        SetDealerCard(trackableName);
        ChangeText.Instance.SetCardText(
            "Player card 1: " + playerCards[0].card
            + "\nPlayer card 2: " + playerCards[1].card
            + "\nDealer card: " + trackableName
            );

        StrategyEnum se = OddsCalculator.Instance.GetStrategy(playerCards, dealerCard);
        switch (se)
        {
            case StrategyEnum.DOUBLE:
                ChangeText.Instance.SetStrategyText("Double!");
                break;
            case StrategyEnum.STAND:
                ChangeText.Instance.SetStrategyText("Stand!");
                break;
            case StrategyEnum.SPLIT:
                ChangeText.Instance.SetStrategyText("Split!");
                break;
            case StrategyEnum.HIT:
                ChangeText.Instance.SetStrategyText("Hit!");
                break;
            case StrategyEnum.SURRENDER:
                ChangeText.Instance.SetStrategyText("Surrender!");
                break;
            case StrategyEnum.UNSUPPORTED:
                ChangeText.Instance.SetStrategyText("UnSupported!");
                break;
            default:
                ChangeText.Instance.SetStrategyText("UnSupported!");
                break;

        }
    }
}
```

Image 13. Method to be called after a card has been recognized. Called only of two or less cards have already been recognized.

The app was usable with most of the cards, but the percentage at which the device recognized some of the lower rated cards was so low, that it would not suffice in a proper project. The cards also had to be levelled almost square in front the device about 20–50cm away, for them to be recognized. When looking at the cards at an angle, e.g. in a situation where the device would have to recognize the cards of a player sitting next to the user, the success ratio was low.

The development of the card counting application finished at this point due to time constraints caused by other projects. The project could be considered an MVP, as the functionality was there, albeit the reliability of image recognition with some of the cards. For a proper project, another way of image recognition should be considered instead of Vuforia's implementation, even though there is no guarantee that others would work better, as some of the cards had only a few recognizable features.

These days many different companies are offering image recognition APIs, most of them offering them for free until X number of requests. Some of the most well-known being Microsoft's Computer Vision API, IBM Watson Visual Recognition, Google Cloud Vision API, Amazon Rekognition and Clarifai. On top of the ones mentioned here, there are many other such services, so there would be plenty of options to replace Vuforia's functionality in the card counter app. [61.]

4.5 Development problems

After building the project solution in Unity and then deploying the app to the device from Visual Studio, it is important do so from the release branch. Building and deploying from the debug branch can make the app take a big hit in performance. During the development of the demo applications, there were cases where an error occurred when trying to deploy from the release branch, but doing so from the debug branch was successful. However, deploying from the debug branch dropped the FPS from 60 to less than 30. After switching back to release branch again, the errors with the deployment were gone.

The development process revealed that using Vuforia in a HoloLens app can cause problems with the usage of HoloLens' camera. The live streaming did not work at all during testing with an app that was using Vuforia. The same problem occurred when trying to take photos while Vuforia was running, but there were no problems while recording video

during the tests. Apparently Vuforia limits real-time access to the camera to guarantee optimal detection and tracking performance. [62.]

5 Conclusion

The purpose of this final year project was to gain knowledge and share knowledge of HoloLens in form of in-depth documentation and demo applications. The research and development of the demo applications accomplished this goal. The weather window application was demonstrated to a customer and the development of the black jack card counter gave insight in to image recognition capabilities with HoloLens, even though more time with the project would have allowed for comparison between different image recognition libraries.

HoloLens has a solid development environment in Unity with MixedRealityToolkit, Vuforia and Visual Studio. Compared to mobile software development, HoloLens 3D application development requires a great deal of configuration in the Unity editor on top of writing code in Visual Studio, whereas when developing an android app, for example, all the work can be done in Android Studio.

Augmented reality seems to be in the beginning of its lifecycle. This can be seen with the phone applications as ARCore and ARKit launched in 2017 and the applications powered by these tools have only been available for about six months. With HoloLens it is clear that the device is still a development edition due to its technical limitations, but the device can nonetheless be seen as a sign of great things to come. The interaction with the device and the tracking of the holograms both work really well, but the form factor as well as the field of view and performance are still holding back the device. Other AR HMDs are in a similar situation, the Meta 2 for example has a high-resolution display and a wide field of view, but it also has the drawback of needing to be connected to a PC.

The maturity of augmented reality headsets can also be seen when looking in to how devices from different manufacturers vary from another. As an example, a standardised way of interaction has not yet been implemented with AR HMDs. Some devices opt for hand gestures, others for controls on the arm of the device or even an external controller.

Some UX guidelines and best practices have taken shape already and Microsoft has done a good job of gathering such resources in their Windows Dev Center. While some of the material is HoloLens specific, some material applies to other devices as well. For example, the way humans expect objects in their vicinity to behave is the same irrespective of device.

Based on what has already been seen with current AR HMDs and the idea of the utility and capabilities that AR can provide, it seems almost inevitable that once the technology matures and becomes more accessible, AR HMDs will be part of people's daily lives as much as smartphones are today.

References

1 Aukstakalnis, Steve. 2016. Practical Augmented Reality: A Guide to the Technologies, Applications, and Human Factors for AR and VR. ebook. <https://books.google.fi/books?id=4oz_DAAAQBAJ>. 1 May 2015. Accessed 4 Oct 2017.

2 Limer, Eric. 2015. How Microsoft's HoloLens Works Its Holographic Magic. Online documentation. <http://www.popularmechanics.com/technology/gadgets/a15324/how-microsofts-hololens-works/>. 1 May 2015. Accessed 4 Oct 2017.

3 Crook, Jordan, Escher, Anna. 2015. A brief history of Snapchat. Online documentation. TechChrunch. <https://techcrunch.com/gallery/a-brief-history-of-snapchat/>. 15 Oct 2015. Accessed 13 Jan 2018.

4 Aslam, Salman. 2018. Snapchat by the Numbers: Stats, Demographics & Fun Facts. Online documentation. Omnicore. <https://www.omnicoreagency.com/snapchat-statistics/>. 1 Jan 2018. Accessed 10 Jan 2018.

5 ZenFone AR. Online documentation. Asus. <https://www.asus.com/us/Phone/ZenFone-AR-ZS571KL/>. Accessed 10 Oct 2017.

6 iPhone X. Online documentation. Apple. <https://www.apple.com/iphone-x/>. Accessed 10 Oct 2017.

7 Introducing ARKit: Augmented Reality for iOS. Online documentation. Apple. <https://developer.apple.com/videos/play/wwdc2017/602/>. Accessed 10 Oct 2017.

8 Kirkpatrick, Ficus. 2017. AR Studio Now in Open Beta, World Effects Rolling Out for Creators. Online documentation. <https://developers.facebook.com/blog/post/2017/12/12/ARStudio-now-in-open-beta/>. 20 Dec 2017. Accessed 22 Dec 2017.

9 Microsoft HoloLens. Online documentation. Microsoft. < https://www.microsoft.com/en-us/hololens>. Accessed 22 Oct 2017.

10 Kastrenakes, Jacob. 2017. Google’s Project Tango is shutting down because ARCore is already here. Online documentation. <https://www.theverge.com/2017/12/15/16782556/project-tango-google-shutting-down-arcore-augmented-reality>. 15 Dec 2017. Accessed 22 Dec 2017.

11 Augment. 2016. Infographic: The History of Augmented Reality. Online documentation. Augment. <http://www.augment.com/blog/infographic-lengthy-history-augmented-reality/>. 12 May 2016. Accessed 10 Jan 2018.

12 Javornik, Ana. 2016. The Mainstreaming of Augmented Reality: A Brief History. Harvard business review. <https://hbr.org/2016/10/the-mainstreaming-of-augmented-reality-a-brief-history>. 4 Oct 2016. Accessed 10 Jan 2018.

13 Virtual Technologies. Online documentation. Volkswagen. <https://www.volkswagenag.com/en/group/research/virtual-technologies.html>. Accessed 10 Oct 2017.

14 Fink, Charlie. 2017. Why Lenovo's 'Star Wars: Jedi Challenge' Is The $200 Device That Will Define AR. Forbes. <https://www.forbes.com/sites/charliefink/2017/11/03/star-wars-jedi-challenge-defines-ar/#7a91d1e81e15>. 3 Nov 2016. Accessed 10 Jan 2018.

15 Sarkar, Samit. 2017. Pokémon Go hits 650 million downloads. Online documentation. Polygon. <https://www.polygon.com/2017/2/27/14753570/pokemon-go-downloads-650-million/>. 27 Feb 2017. Accessed 10 Jan 2018.

16 Lee, Dami. 2017. Ikea Place is an AR app that lets you put furniture on the street. Online documentation. The Verge. <https://www.theverge.com/2017/9/20/16339006/apple-ios-11-arkit-ikea-place-ar-app>. 20 Sep 2017. Accessed 10 Jan 2018.

17 Koetsier, John. 2017. The Chalk AR App Is The Future Of Education, Service And Support. Online documentation. Forbes. <https://www.forbes.com/sites/johnkoetsier/2017/10/03/the-chalk-ar-app-is-the-future-of-education-service-and-support/#207e9eea2505>. 3 Oct 2017. Accessed 14 Jan 2018.

18 Fingas, Jon. 2017. Edmunds uses AR to show how a new car will fit in your garage. Online documentation. Forbes. <https://www.engadget.com/2017/09/19/edmunds-augmented-reality-car-fit-feature/>. 19 Sep 2017. Accessed 10 Jan 2018.

19 HOW DOES TAPMEASURE WORK?. Online documentation. Occipital. <https://tapmeasure.io/help/how-does-tapmeasure-work>. Accessed 10 Jan 2018.

20 The Meta 2: Made for AR App Development. Online documentation. Meta. <https://meta-eu.myshopify.com/>. Accessed 26 Dec 2018.

21 Baldassi, Stefano. 2016. The Natural Machine. Online documentation. Meta Vision. <https://blog.metavision.com/the-natural-machine>. Accessed 9 Jan 2018.

22 Odom, Jason. 2017. A Hands-On with the Meta 2 Head-Mounted Display. Online documentation. Next Reality. <https://meta.reality.news/news/hardware-review-hands-with-meta-2-head-mounted-display-0178607/>. 19 Jul 2017. Accessed 6 Nov 2017.

23 Grant, Rebecca. 2014. Mysterious startup Magic Leap raises $50M to build a 'rocket ship for the mind'. Online documentation. Venture Beat. <https://venturebeat.com/2014/02/05/mysterious-startup-magic-leap-raises-50m-to-build-a-rocket-ship-for-the-mind/>. 5 Feb 2014. Accessed 6 Nov 2017.

24 Leswing, Kif. 2017. It's official: Magic Leap adds another $502 million to its war chest. Online documentation. Business Insider. <http://nordic.businessinsider.com/magic-leap-series-d-official-502-million-2017-10?r=US&IR=T>. 17 Oct 2017. Accessed 6 Nov 2017.

25 Robertson, Adi. 2017. Magic Leap finally unveils augmented reality goggles, says it's shipping next year. Online documentation. The Verge. <https://www.theverge.com/2017/12/20/16800474/magic-leap-one-creator-edition-augmented-reality-goggles-announce>. 20 Dec 2017. Accessed 20 Dec 2017.

26 R9 Tech sheet. Online documentation. ODG. <https://www.osterhoutgroup.com/pub/static/version1511977599/frontend/Infortis/ultimo/en_US/pdf/R-9-TechSheet.pdf>. Accessed 6 Nov 2017.

27 Lang, Ben. 2017. ODG Announces Two New Smartglasses With Positional Tracking, Expanded Field of View. Online documentation. Road to VR. <https://www.roadtovr.com/odg-announces-smart-glasses-r8-r9-ar-qualcomm-snapdragon-835/>. 3 Jan 2017. Accessed 27 Dec 2017.

28 Odom, Jason. 2017. ODG's New R-7HL Are the First Rugged Smartglasses Made Specifically for the industrial Workforce. Online documentation. Next Reality. <https://augmented.reality.news/news/odgs-new-r-7hl-are-first-rugged-smart-glasses-made-specifically-for-industrial-workforce-0176925/>. 5 Apr 2017. Accessed 10 Jan 2018.

29 Moverio BT-350 Smart Glasses. Online documentation. Epson. <https://epson.com/For-Work/Wearables/Smart-Glasses/Moverio-BT-350-Smart-Glasses/p/V11H837020>. Accessed 10 Jan 2018.

30 Kothari, Jay. 2017. A new chapter for Glass. Online documentation. X Development. <https://blog.x.company/a-new-chapter-for-glass-c7875d40bf24>. Accessed 10 Jan 2018.

31 Glass. Online documentation. X Development. <https://x.company/glass/>. Accessed 10 Jan 2018.

32 The Vuzix Blade™ Augmented Reality Smart Glasses for Enterprise. Online documentation. Vuzix. <https://www.vuzix.com/Products/Blade-Enterprise>. Accessed 8 Jan 2018.

33 DreamWorld. Online documentation. <http://www.dreamworldvision.com/>. Accessed 4 Nov 2017.

34 Join the resolution revolution. Online documentation. Varjo. <https://varjo.com/early-access/>. Accessed 8 Jan 2018.

35 Rubino, Daniel. 2016. Microsoft HoloLens - Here are the full processor, storage and RAM specs. Online documentation. <https://www.windowscentral.com/microsoft-hololens-processor-storage-and-ram>. 2 May 2016. Accessed 4 Oct 2017.

36 Windows Mixed Reality – Comfort. Online documentation. Microsoft. <https://developer.microsoft.com/en-us/windows/mixed-reality/comfort>. Accessed 10 Oct 2017.

37 Rastas, Ossi. Passionate software developer, Vincit Development Oy, Helsinki. User testing. 28 Aug 2017.

38 Gaze. Online documentation. Microsoft. <https://developer.microsoft.com/en-us/windows/mixed-reality/gaze>. Accessed 10 Oct 2017.

39 Windows Mixed Reality – Gestures. Online documentation. Microsoft. <https://developer.microsoft.com/en-us/windows/mixed-reality/gestures>. Accessed 4 Oct 2017.

40 Windows Mixed Reality – Voice input. Online documentation. Microsoft. <https://developer.microsoft.com/en-us/windows/mixed-reality/voice_input>. Accessed 4 Oct 2017.

41 Coordinate systems. Online documentation. Microsoft. <https://developer.microsoft.com/en-us/windows/mixed-reality/coordinate_systems>. Accessed 10 Oct 2017.

42 Spatial Anchors. Online documentation. Microsoft. <https://developer.microsoft.com/en-us/windows/mixed-reality/spatial_anchors>. Accessed 10 Oct 2017.

43 Ashley, James. 2016. Understanding HoloLens spatial mapping and hologram ranges. Online documentation. <http://www.imaginativeuniversal.com/blog/2016/03/23/understanding-hololens-spatial-mapping-and-hologram-ranges/>. 23 Mar 2016. Accessed 10 Oct 2017.

44 More than 1200 triangles per qubic meter. Forum discussion. Microsoft. <https://forums.hololens.com/discussion/1873/more-than-1200-triangles-per-qubic-meter>. Accessed 10 Oct 2017.

45 Spaces on HoloLens. Online documentation. Microsoft. <https://support.microsoft.com/lo-la/help/13760/hololens-spaces-on-hololens>. Accessed 11 Oct 2017.

46 Hologram stability. Online documentation. Microsoft. <https://developer.microsoft.com/en-us/windows/mixed-reality/hologram_stability>. Accessed 10 Oct 2017.

47 Lamkin, Paul. 2017. Best VR headsets 2018: HTC Vive, Oculus, PlayStation VR compared. Online documentation. Wearable. <https://www.wareable.com/vr/best-vr-headsets-2017>. 18 Dec 2017. Accessed 27 Dec 2017.

48 Sharing. Online documentation. Microsoft. <https://github.com/Microsoft/MixedRealityToolkit-Unity/wiki/Sharing>. Updated 6 Oct 2017. Accessed 22 Oct 2017.

49 Van Amerongen, Robbrecht. 2016. How to run a demo with Microsoft Hololens and share your screen. Online documentation. AMIS Technology blog. <https://technology.amis.nl/2016/12/05/demo-microsoft-hololens-and-share-your-screen/>. 5 Jun 2016. Accessed 27 Dec 2017.

50 Meijers, Alexander. 2017. Importing 3D models into Unity for HoloLens applications. Online documentation. AppzInside. <http://www.appzinside.com/2017/05/27/importing-3d-models-into-unity-for-hololens-applications/>. 27 May 2017. Accessed 14 Jan 2018.

51 Patrizio, Andy. 2017. Japan Airlines employs Microsoft HoloLens for inspections and training. Online documentation. Network World. <https://www.networkworld.com/article/3098505/software/japan-airlines-employs-microsoft-hololens-for-inspections-and-training.html>. 22 Jul 2016. Accessed 14 Jan 2018.

52 Landau, Elizabeth. 2016. 'Mixed Reality' Technology Brings Mars to Earth. Online documentation. Nasa. <https://www.nasa.gov/feature/jpl/mixed-reality-technology-brings-mars-to-earth>. 30 Mar 2016. Accessed 14 Jan 2018.

53 Microsoft HoloLens. Online documentation. Volvo. <https://www.volvocars.com/uk/about/humanmade/projects/hololens>. Accessed 15 Jan 2018.

54 Fragments. Online documentation. Microsoft. <https://www.microsoft.com/en-us/hololens/apps/fragments>. Accessed 4 Jan 2018.

55 Case study - Expanding the spatial mapping capabilities of HoloLens. Online documentation. Microsoft. <https://developer.microsoft.com/en-us/windows/mixed-reality/case_study_-_expanding_the_spatial_mapping_capabilities_of_hololens>. Accessed 4 Jan 2018.

56 Heinonen, Eemeli. Passionate junior software developer, Vincit Development Oy, Espoo. Development and testing. 30 Aug 2017.

57 Windows Mixed Reality – Install the tools. Online documentation. Microsoft. <https://developer.microsoft.com/en-us/windows/mixed-reality/Install_the_tools.html#installation_checklist_for_immersive_headsets>. Accessed 27 Dec 2017.

58 Buildin 2D apps. Online documentation. Microsoft. <https://developer.microsoft.com/en-us/windows/mixed-reality/building_2d_apps>. Accessed 4 Jan 2018.

59 HoloLens: Building UWP 2D Apps for Microsoft HoloLens. Online documentation. Channel 9. <https://channel9.msdn.com/Events/Build/2016/B854>. Accessed 4 Jan 2018.

60 What is MixedRealityToolkit-Unity?. Online documentation. Microsoft. <https://github.com/Microsoft/MixedRealityToolkit-Unity>. Updated 20 Oct 2017. Accessed 22 Oct 2017.

61 Domes, Scott. 2017. We compared the 3 best image analysis API's—here's what we learned. Online documentation. Musefind. <https://engineering.musefind.com/we-compared-the-3-best-image-analysis-apis-here-s-what-we-learned-2d54cff5ae62>. Accessed 6 Jan 2018.

62 Video Recording and Photo taking not working with Hololens. Online documentation. <https://developer.vuforia.com/forum/hololens/video-recording-and-photo-taking-not-working-hololens>. Accessed 28 Dec 2017.

SimpleJSON example

Examples (C# / UnityScript)

This is the JSON string which will be used in this example:

```
{
    "version": "1.0",
    "data": {
        "sampleArray": [
            "string value",
            5,
            {
                "name": "sub object"
            }
        ]
    }
}
```

```
var N = JSON.Parse(the_JSON_string);
var versionString = N["version"].Value;         // versionString will be a string containing "1.0"
var versionNumber = N["version"].AsFloat;       // versionNumber will be a float containing 1.0
var name = N["data"]["sampleArray"][2]["name"];// name will be a string containing "sub object"

//C#
string val = N["data"]["sampleArray"][0];       // val contains "string value"

//UnityScript
var val : String = N["data"]["sampleArray"][0];// val contains "string value"

var i = N["data"]["sampleArray"][1].AsInt;      // i will be an integer containing 5
N["data"]["sampleArray"][1].AsInt = i+6;        // the second value in sampleArray will contain "11"

N["additional"]["second"]["name"] = "FooBar";  // this will create a new object named "additional" in this object create another
                                                //object "second" in this object add a string variable "name"

var mCount = N["countries"]["germany"]["moronCount"].AsInt; // this will return 0 and create all the required objects and
                                                            // initialize "moronCount" with 0.

if (N["wrong"] != null)                         // this won't execute the if-statement since "wrong" doesn't exist
{}
if (N["wrong"].AsInt == 0)                      // this will execute the if-statement and in addition add the "wrong" value.
{}

N["data"]["sampleArray"][-1] = "Test";          // this will add another string to the end of the array
N["data"]["sampleArray"][-1]["name"] = "FooBar"; // this will add another object to the end of the array which contains a string named "name"

N["data"] = "erased";                           // this will replace the object stored in data with the string "erased"
```

Blackjack strategy table

	ADVANCED BLACKJACK STRATEGY TABLE									
	Dealer's First Card									
Your Hand	2	3	4	5	6	7	8	9	10	A
18+	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND
17	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND
16	STAND	STAND	STAND	STAND	STAND	HIT	HIT	HIT	HIT	HIT
15	STAND	STAND	STAND	STAND	STAND	HIT	HIT	HIT	HIT	HIT
14	STAND	STAND	STAND	STAND	STAND	HIT	HIT	HIT	HIT	HIT
13	STAND	STAND	STAND	STAND	STAND	HIT	HIT	HIT	HIT	HIT
12	HIT	HIT	STAND	STAND	STAND	HIT	HIT	HIT	HIT	HIT
11	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	HIT
10	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	HIT	HIT
9	HIT	DOUBLE	DOUBLE	DOUBLE	DOUBLE	HIT	HIT	HIT	HIT	HIT
8	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT
7	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT
6	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT
5	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT	HIT
Soft 20	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND
Soft 19	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND
Soft 18	STAND	DOUBLE	DOUBLE	DOUBLE	DOUBLE	STAND	STAND	HIT	HIT	HIT
Soft 17	HIT	DOUBLE	DOUBLE	DOUBLE	DOUBLE	HIT	HIT	HIT	HIT	HIT
Soft 16	HIT	HIT	DOUBLE	DOUBLE	DOUBLE	HIT	HIT	HIT	HIT	HIT
Soft 15	HIT	HIT	DOUBLE	DOUBLE	DOUBLE	HIT	HIT	HIT	HIT	HIT
Soft 14	HIT	HIT	HIT	DOUBLE	DOUBLE	HIT	HIT	HIT	HIT	HIT
Soft 13	HIT	HIT	HIT	DOUBLE	DOUBLE	HIT	HIT	HIT	HIT	HIT
Pair A	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT
Pair 10	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND	STAND
Pair 9	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	STAND	SPLIT	SPLIT	STAND	STAND
Pair 8	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT
Pair 7	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	HIT	HIT	HIT	HIT
Pair 6	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	HIT	HIT	HIT	HIT	HIT
Pair 5	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	DOUBLE	HIT	HIT
Pair 4	HIT	HIT	HIT	SPLIT	SPLIT	HIT	HIT	HIT	HIT	HIT
Pair 3	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	HIT	HIT	HIT	HIT
Pair 2	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	SPLIT	HIT	HIT	HIT	HIT