Bacheloroppgave

IBE600 Bacheloroppgave

Title: Object Detection with HoloLens 2 using Mixed Reality and Unity a proof-of-concept

Forfatter

Simen Stokkeland Fuglseth

Totalt antall sider inkludert forsiden: 42

Molde, Innleveringsdato

25.05.2022

Obligatorisk egenerklæring/gruppeerklæring

Den enkelte student er selv ansvarlig for å sette seg inn i hva som er lovlige hjelpemidler, retningslinjer for bruk av disse og regler om kildebruk. Erklæringen skal bevisstgjøre studentene på deres ansvar og hvilke konsekvenser fusk kan medføre. Manglende erklæring fritar ikke studentene fra sitt ansvar.

Du/dere fyller ut erklæringen ved å klikke i ruten til høyre for den enkelte del 1-6:		
1.	**Jeg/vi erklærer herved at min/vår besvarelse er mitt/vårt eget arbeid, og at jeg/vi ikke har brukt andre kilder eller har mottatt annen hjelp enn det som er nevnt i besvarelsen.**	☒
2.	**Jeg/vi erklærer videre at denne besvarelsen:** • ikke har vært brukt til annen eksamen ved annen avdeling/universitet/høgskole innenlands eller utenlands. • ikke refererer til andres arbeid uten at det er oppgitt. • ikke refererer til eget tidligere arbeid uten at det er oppgitt. • har alle referansene oppgitt i litteraturlisten. • ikke er en kopi, duplikat eller avskrift av andres arbeid eller besvarelse.	☒
3.	**Jeg/vi er kjent med at brudd på ovennevnte er å betrakte som fusk og kan medføre annullering av eksamen og utestengelse fra universiteter og høgskoler i Norge, jf. Universitets- og høgskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§16 og 36.**	☒
4.	**Jeg/vi er kjent med at alle innleverte oppgaver kan bli plagiatkontrollert, jf. høgskolens regler og konsekvenser for fusk og plagiat**	☒
5.	**Jeg/vi er kjent med at høgskolen vil behandle alle saker hvor det forligger mistanke om fusk etter høgskolens retningslinjer for behandling av saker om fusk**	☒
6.	**Jeg/vi har satt oss inn i regler og retningslinjer i bruk av kilder og referanser på biblioteket sine nettsider**	☒

Personvern

Personopplysningsloven

Forskningsprosjekt som innebærer behandling av personopplysninger iht. Personopplysningsloven skal meldes til Norsk senter for forskningsdata, NSD, for vurdering.

Har oppgaven vært vurdert av NSD? ☐**ja** ☒**nei**

- Hvis ja:

Referansenummer:

- Hvis nei:

Jeg/vi erklærer at oppgaven ikke omfattes av Personopplysningsloven: ☒

Helseforskningsloven

Dersom prosjektet faller inn under Helseforskningsloven, skal det også søkes om forhåndsgodkjenning fra Regionale komiteer for medisinsk og helsefaglig forskningsetikk, REK, i din region.

Har oppgaven vært til behandling hos REK? ☐**ja** ☒**nei**

- Hvis ja:

Referansenummer:

Publiseringsavtale

Studiepoeng: 15

Veileder: Judith Ann Molka-Danielsen

Fullmakt til elektronisk publisering av oppgaven

Forfatter(ne) har opphavsrett til oppgaven. Det betyr blant annet enerett til å gjøre verket tilgjengelig for allmennheten (Åndsverkloven. §2).

Alle oppgaver som fyller kriteriene vil bli registrert og publisert i Brage HiM med forfatter(ne)s godkjennelse.

Oppgaver som er unntatt offentlighet eller båndlagt vil ikke bli publisert.

Jeg/vi gir herved Høgskolen i Molde en vederlagsfri rett til å gjøre oppgaven tilgjengelig for elektronisk publisering:	☒**ja**	☐**nei**
Er oppgaven båndlagt (konfidensiell)?	☐**ja**	☒**nei**
(Båndleggingsavtale må fylles ut)		
- Hvis ja:		
Kan oppgaven publiseres når båndleggingsperioden er over?	☐**ja**	☐**nei**

Dato:

Antall ord: 6200

Forord / Acknowledgments

This marks the end of three years of a bachelor's degree in IT og digitalisering at Høgskolen i Molde. Even though this study period has been under pressure from a pandemic and with little physical studying, I would like to thank those who have supported me. A big thanks to Axbit, who gave me the task on which my thesis is based and also provided me with a HoloLens 2 for me to work with. I would also like to thank Andreas Reiten, my mentor from Axbit, who gave me guidance throughout my thesis. Then I would like to thank Judith Ann Molka-Danielsen, who has been my supervisor.
Finally, I would like to thank all of my family and friends who have also supported me.

Abstract

The primary objective of this thesis is to be a proof of concept on how to do object detection and use Augmented and Mixed Reality to position a holographic label anchored in the physical world. This can then be the groundwork for using object detection with specific use cases in mind for the future.

With Microsoft HoloLens 2, an Mixed Reality headset, the idea was implemented using Microsoft Azure Custom Vision for object detection. To create the application, Unity was used as the development platform and certain helper tools such as Mixed Reality Tool Kit for Unity.

This project developed a methodology and application that trains a Machine Learning deep learning model to recognize objects using object detection through a Mixed Reality headset. The model was trained images from an open-source dataset to recognize a sample set of objects. The methodology and application were trialed, and the sample set of objects was successfully recognized.

Norwegian Abstract

Hovedmålet med denne oppgaven er å være et proof of concept om hvordan man kan gjøre gjenstandsdeteksjon og bruke Augmented og Mixed Reality for å posisjonere et objekt i den fysiske verden. Dette kan da være grunnlaget for bruk av objektdeteksjon med spesifikke brukstilfeller i tankene for fremtiden.

Med Microsoft HoloLens 2, et Augmented Reality-hodesett, ble ideen implementert ved å bruke Microsoft Azure Custom Vision for objektdeteksjon. For å lage applikasjonen ble Unity brukt som utviklingsplattform og visse hjelpeverktøy som Mixed Reality Tool Kit for Unity.

Dette prosjektet utviklet en metodikk og applikasjon som trener en maskinlæringsmodell for dyp læring til å gjenkjenne objekter ved å bruke objektdeteksjon gjennom et Mixed Reality-headset. Modellen ble trent bilder fra et åpen kildekode-datasett for å gjenkjenne et eksempelsett med objekter. Metodikken og anvendelsen ble utprøvd, og prøvesettet med objekter ble gjenkjent.

Content

List of Figures

Acronyms

AG Augmented Reality

AI Artificial Intelligence

API Application Programming Interface

CNN Convolutional Neural Networks

COCO Common Objects in Context

GPU Graphic Processing Unit

ML Machine Learning

MR Mixed Reality

R-CNN Region-based Convolutional Neural Network

SDK Software Developing Kits

UWP Universal Windows Platform

YOLO You Only Look Once

1.0 Introduction

Object detection has a wide variety of use cases and has received much attention in recent years from researchers and others. With the development of deep learning, more powerful tools, and better understanding, the accuracy and variety of objects that can be detected will only increase in the future. Some fields where object detection could be essential include healthcare, security, agriculture, autonomous driving, and facial recognition.

First, what is meant by object detection must be clarified. Object detection is important within Computer Vision and uses classification and localization determined from an image. The predicted class of one or more objects in an image is referred to as *image classification*. *Object localization* is the process of locating one or more objects in an image and determining the bounding box extent of the objects identified. *Object detection* integrates these two tasks by locating and classifying one or more objects in an image. (Jason Brownlee. 2019.) Computer Vision is the field of how computers can gain high-level understanding from images or videos.

Some use cases where object detection could be necessary include autonomous driving, people detection for security, and medical detection for healthcare.

For this thesis, the physical/real world will be referred to as the physical world and the virtual/digital world as the virtual world. The traditional definition of a Virtual Reality (VR) environment is one in which the participant-observer is completely immersed in and can interact with an entirely computer-generated world. (MILGRAM and KISHINO 1994) In VR, the user feels like the computer-generated environment is real and immersive. The experience can be similar or different from the physical world, but VR contains total immersion in any case. Mixed Reality (MR), a subset of VR, merges the virtual world with the physical world. A way to merge the physical and virtual worlds is through computer-generated inputs, which enhance the user's physical world. The merging of the physical and virtual world is called Augmented Reality (AR).

While a VR headset will give the user total immersion by completely covering their field of view, AR and MR headsets enhance the user's experience and add computer-generated objects to the user's field of view. The user can also get AR and MR experience with just a phone, such as Pokemon Go. However, this experience does not have the same potential as a dedicated headset and will not be as immersive. An MR headset such as HoloLens 2

provides the user with an enhanced experience by creating visual holographic in the user's field of view and audio. Also, it provides a method for the user to interact with their surroundings, either through speech commands or gestures.
Machine Learning (ML) is another subject receiving more attention in recent years since it is used within Computer Vision and also other scientific fields due to the technology and understanding becoming more available. In particular deep learning, a part of Neural Networks (NN), is growing fast and is what is usually used in object detection. Some examples are already mentioned above about object detection, but also everyday devices such as Alexa and Siri use some form of ML.

1.1 Structure of the thesis

Chapter one - Introduction
Introduces the project and presents the motivation and goal of the thesis.

Chapter two - Background
Contains topics related to the project's background, such as the applied technologies, subjects, frameworks, and related works.

Chapter three - Proposed Method
Has a proposed method on how to solve the problem stated in the goal. The proposed method uses the applied technologies, subjects, and frameworks mentioned in chapter two.

Chapter four - Evaluation
This chapter has an experiment and discusses the results from the experiment as well as how others can replicate this.

Chapter five - Discussion
A discussion about the design and implementation chosen in the thesis and what alternatives were considered.

Chapter six- Conclusion and Future Outlook

Concludes if the proposed method solved the goal and discusses the future outlook.

1.2 Motivation

With digital transformation, companies, and government institutions are looking for new ways to use digital technology to help create innovation, value, or otherwise become a more efficient business or institution. Recently, there has also been much buzz around the metaverse. The Facebook company even changed its name to Meta announced 28.10.2021 in anticipation that the metaverse will be the next digital frontier. The metaverse is the post-reality universe that combines physical reality and digital virtuality in a continual and persistent multiuser environment. It is centered on the convergence of VR and AR technologies that enable multisensory interactions with virtual environments, digital objects, and people. (Mystakidis 2022)

MR is still a relatively new emerging technology but still has great potential in what can soon be possible. The blending of the physical world with the digital world will bring about exciting new possibilities and technologies that will digitally transform workplaces. Some of the fields that could use MR in the future could be education, construction, and healthcare, to mention a few. HoloLens 2 could, as an example, be used by a doctor who is doing surgery in a remote location while getting guidance from a specialist watching the video feed. Another example could be in construction to help visualize how a room or a building will look and use that to make design changes. MR can also make the workplace safer with the ability to visualize from a distance or show a worker part of the site which is off-limits due to danger.

MR headsets also bring other benefits that might change how work and free time are spent. However, one limitation with screens is that everything that is going on has to happen on the screen itself. An MR headset could perhaps have multiple tabs open from a web browser and view them all while being more limited by the physical room itself. For example, having multiple screens takes space, while a virtual screen does not take more space than the headset itself.

It is in the author's view that open and free access to information greatly benefits everyone involved and should be embraced as often as possible. Without open-source information and software, much of this project would not be possible therefore, the source code for this project will also be open-source and found on GitHub.

1.3 Goal

This Bachelor thesis aims to create a proof of concept program using HoloLens 2 that enables object detection using ML, AR, and MR. Object recognition should be done in near-real-time by an ML algorithm that has been trained to recognize specific objects, such as a knife, cup, banana, etc. The detection process should start after the user enables it through an input such as voice input or gesture. Once an object has been recognized, the position should be able to be used to visualize a text graphic (2D or 3D) using MR displayed in the field of view, close to the real-world position of the object. To do the visualization, an MR headset will be used, which in this case will be a HoloLens 2.

2.0 Background

This chapter will describe the tools and technologies used to implement the goals of this thesis. It will also go through the dataset used and related work that mentions other works relevant to this subject.

2.1 Technologies

The main object needed to complete this project is some sort of MR headset. In this case, it will be Microsoft HoloLens 2. To develop for the HoloLens 2, Unity will be used as the developing platform. Mixed Reality Toolkit for Unity is a tool for helping developers with MR and AR and is made by Microsoft. To do object detection, Microsoft Azure Custom Vision, a cloud service, will be applied.

2.1.1 Microsoft HoloLens 2

The primary technology in this project is an MR headset from Microsoft called HoloLens 2. HoloLens 2 is an untethered holographic computer that provides the user with an MR experience that runs on Windows 10. It does this using holographic lenses and holographic processing that can blend the physical world with a simulated graphic or a virtual world. HoloLens 2 is the successor to the HoloLens 1 and provides a more comfortable and immersive experience than its predecessor.

Figure 2.1: HoloLens 2. (Microsoft. 2019.)

Table 2.1: HoloLens 2 Technical specifications and capabilities (Microsoft. 2019.)

Display	**Optics:** See-through holographic lenses (waveguides) **Resolution:** 2k 3:2 light engines **Holographic Density:** >2.5k radiants (light points per radian) **Eye-based Rendering:** Display optimization for 3D eye position
Sensors & Audio	**Depth:** 1-MP time-of-flight (ToF) depth sensor **IMU:** Accelerometer, gyroscope, magnetometer **Camera:** 8MP stills, 1080p 30fps video **Microphone Array:** 5 channels **Speakers:** Built-in, spatial audio
Human Understanding	**Hand Tracking:** Two-handed fully articulated model, direct manipulation **Eye Tracking:** Real-time tracking **Voice:** Command and control on-device, Natural Language with

	internet connectivity
Environmental Understanding	**6DoF Tracking:** World-scale positional tracking **Spatial Mapping:** Real-time environment mesh **Mixed Reality Capture:** Mixed hologram and physical environment photos and videos
Computer & Connectivity	**SoC (system on a chip):** Qualcomm Snapdragon 850 Compute Platform **HPU:** 2nd Generation Custom-built Holographic Processing Unit **Wi-Fi:** 802.11ac 2x2 **Bluetooth:** 5.0 **USB:** USB Type-C

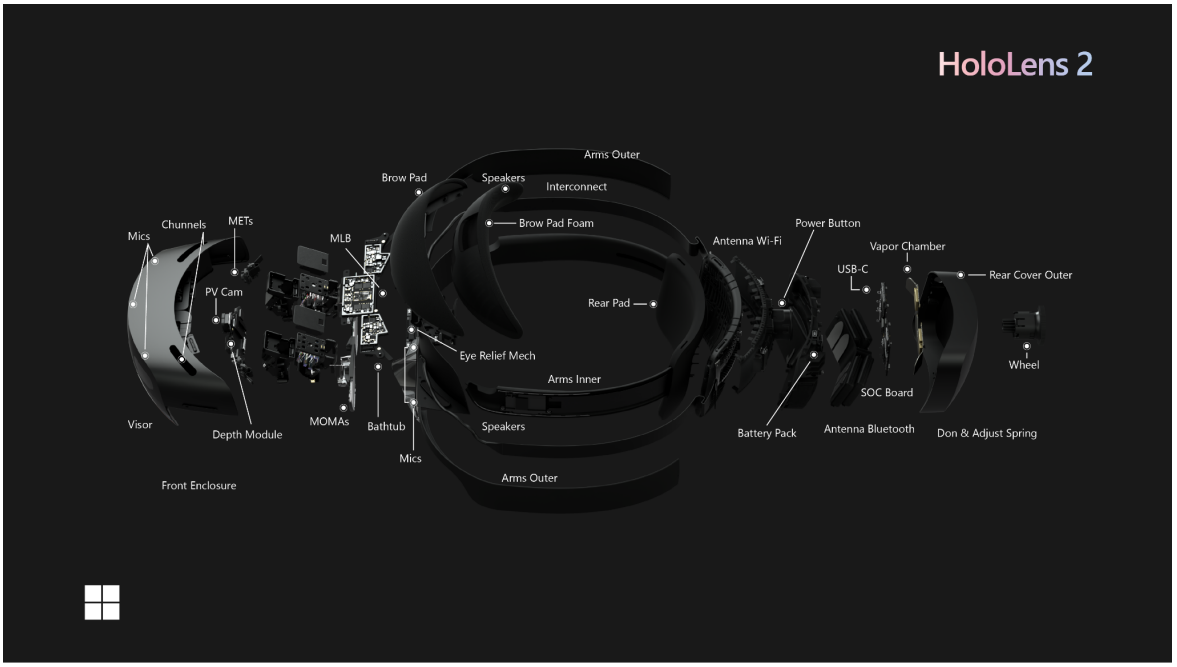

Figure 2.2: HoloLens 2 components (Scooley. 2020.)

The HoloLens 2 contains a variety of optical sensors, including four head-tracking cameras, two on each side, for peripheral environment perception. A near- and far-range depth camera detects hand motions, and specialized speakers simulate sound from any location in the room. There are also many microphones, an HD camera in the headset's center, and an ambient light sensor. The sensors and cameras work together to detect and track the room's

floor, walls, and major objects. Spatial awareness is a technique for blending holograms into their surroundings. (“7 Things about Microsoft Hololens,” n.d.) The battery life is around 2-3 hours of active use. The HoloLens 2 can interact with the MR experience either through voice commands or input gestures from the user example of input gesture in *figure 2.3*.

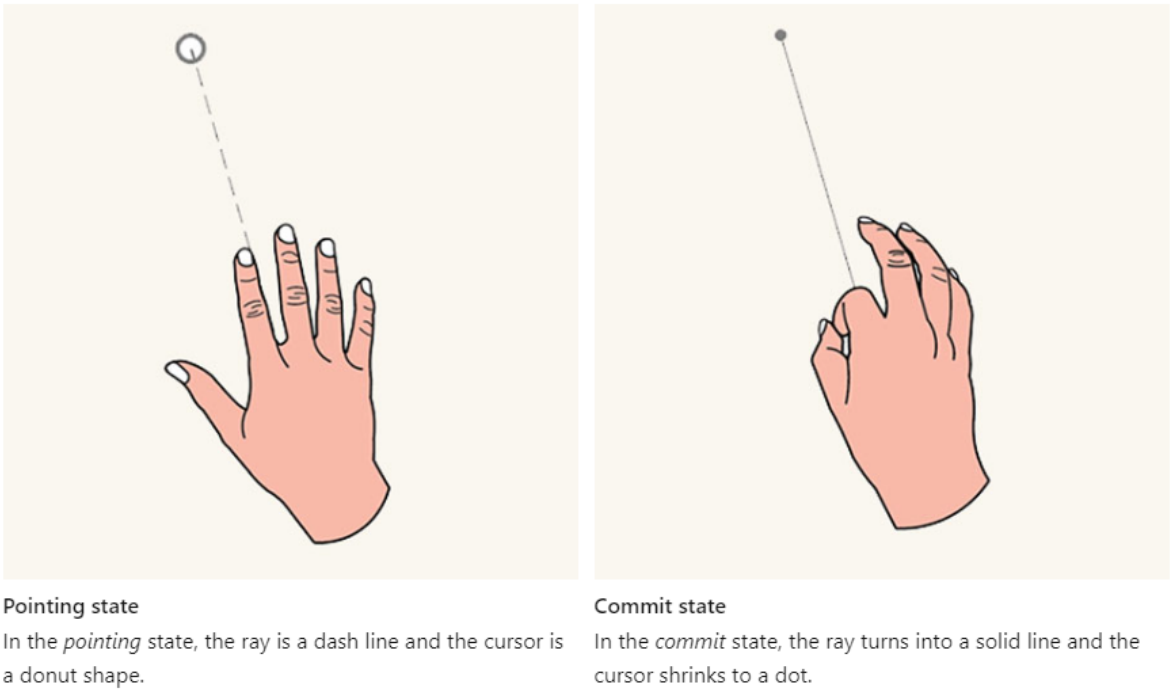

Figure 2.3: Example of hand rays pointing and commit state. (Caseymeekhof. 2022.)

2.1.2 Object Detection

When it comes to object detection, the methods generally fall into two categories neural network-based or non-neural approaches. However, the most common method used is the neural network-based method since they do not require features to be defined. Neural network-based methods are typically based on Convolutional Neural Networks (CNN). Some commonly used approaches include You Only Look Once (YOLO) and Region proposals such as Region-based Convolutional Neural Networks. (R-CNN)
As mentioned in the introduction, one needs to combine image classification and object localization to do object detection. Image classification to get one or multiple classes from the image, and object localization to draw a bounding box for the location of the objects in the image.

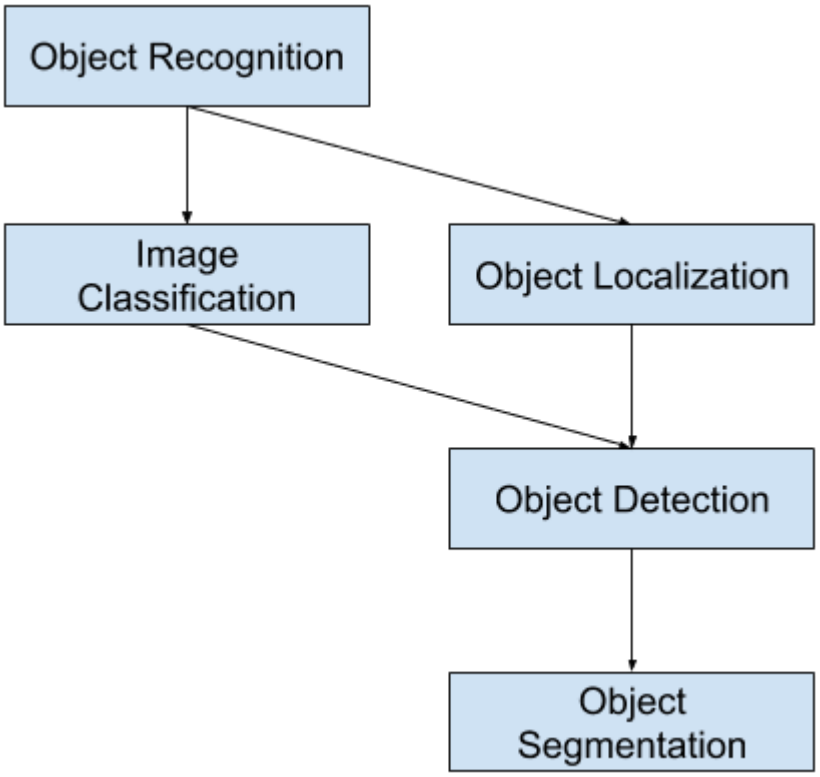

Figure 2.4: Overview of Object Recognition Computer Vision Tasks. (Jason Brownlee. 2019.)

2.1.3 Machine Learning

ML is concerned with how to create computers that improve themselves over time through experience. It is one of today's fastest expanding technical topics, located at the crossroads of computer science and statistics and at the core of artificial intelligence and data science. (Horvitz and Mulligan 2015)

Computer Vision contains all that has to do with computers gaining high-level understanding from images and videos, including object detection and object recognition. For example, AI deep learning methods such as CNN are usually used since it is generally less time-consuming by not needing to define features for object detection and are also more accurate.

NN and deep learning have fundamentally altered the way we approach object detection. Object detection performance improved dramatically with the introduction of the YOLO and R-CNN families. (Mokhtari. 2021.)

Knowing this, the most apparent solution to object detection is using ML and, in particular, deep learning to train a model that uses object detection. There are multiple methods for how this can be done, either by creating an ML model or using an external cloud service to train the ML model like Microsoft Azure Custom Vision. Some benefit to using a service such as Microsoft Azure Custom Vision is that an external GPU is used for training and later detection. Alternatively, the model can be exported, and then the model can be used without being connected to WIFI.

2.1.4 Microsoft Azure Custom Vision

Azure Custom Vision is a component of Azure Cognitive Services, a cloud-based Artificial Intelligence (AI) service that assists developers in incorporating cognitive intelligence into their applications. REST Application Programming Interface (API), client library Software Developing Kits (SDK), and user interfaces are all available. (Nitinme. 2022.) Azure Custom Vision is an image recognition service that can create, deploy, and improve image identifier models. An image identifier labels images based on the objects detected. Each label represents a classification or an object. Custom vision analyzes images with ML and builds a model based on the images used for training. The algorithm then tests itself on the images to calculate accuracy, dividing the data into training, validation, and testing. Custom vision can be used to classify images as well as to detect objects. Images and labeling for training can be submitted either manually or via API. Custom vision does not require a large number of images to recognize major differences between images, but it is not ideal for detecting subtle differences between images. (PatrickFarley. 2022.) With Custom Vision, the created model can be used with an API, or the model can be exported, enabling the model to be used without using the API. Since Custom Vision is a cloud service, this takes away some of the computational requirements of using ML for object detection.

2.1.5 Mixed Reality

Although the idea of MR has been around for some time, it is not until more recently that the technology requirements to truly make an immersive experience have become more

readily available. The most common and mainstream use of AR today is mobile AR on social media with, for example, filters, which people might not realize is a form of AR. In Virtual Reality (VR), the entire world is computer-generated, and a user requires a VR headset that completely covers their field of view. MR is the blend of the physical and virtual worlds where holographs or even larger and more complex virtual worlds are added on top of reality. MR-headsets such as HoloLens can make the experience more immersive with environmental understanding, hand tracking, eye tracking, speech input, spatial sound, location, and positioning in physical and virtual worlds. Mobile phones do not have as many or none of the same environmental perception capabilities.

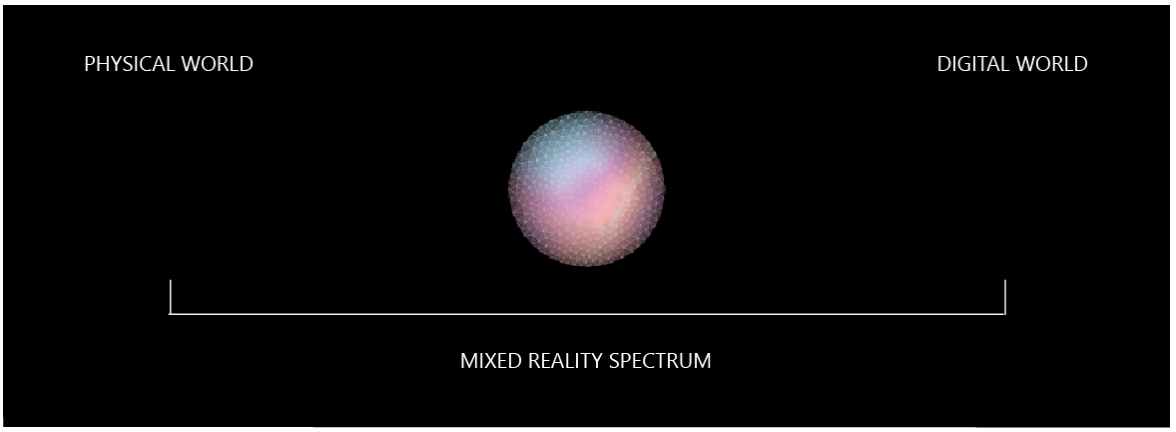

Figure 2.5: Mixed reality is the spectrum between the physical and digital world. (BrandonBray. 2022.)

The concept of a "virtuality continuum" refers to the mixture of classes of objects presented in any given display situation, with real environments at one end of the spectrum and virtual environments at the other end of the spectrum. (MILGRAM and KISHINO 1994)
On the MR spectrum/virtuality continuum, AR falls more towards the physical world end of the spectrum with a view of the physical world where holographic visuals, sound, or any other form of enhancement of the senses has been added to the experience.

2.1.6 Unity

Unity is a cross-platform game engine developed by Unity Technologies. It is the leading development platform for developing 2D and 3D games. Scripting is done with C#, and it has built-in Visual Studio integration. Unity also has a lot of plugins and packages that can

help with developing in Unity, such as Mixed Reality Tool Kit. (MRTK) Unity as a developer also provides a fast way to prototype with the *play* button, enabling the project to run without building the project. Unity supports HoloLens through Universal Windows Platform (UWP), which creates client applications that run on Windows devices. Since Unity is a cross-platform engine, it is made to be able to use the same program with minimal changes across multiple platforms and devices in mind. This is quite useful since there exist many different MR headset devices on the market today.

2.1.7 Mixed Reality Tool Kit

MRTK for Unity is a Microsoft-led project that provides a set of components and features that can be used to speed up cross-platform MR app development in Unity. The following are some of its functions:

- Provides the cross-platform input system and building blocks for spatial interactions and UI.
- Enables rapid prototyping via in-editor simulation that allows the user to see changes immediately.
- It operates as an extensible framework that provides developers the ability to swap out core components.
- Supports a wide range of platforms, including OpenXR (Unity 2020.3.8+) and Windows MR.

(Polar-Kev. 2022.)

2.2 Dataset

The proposed method for object detection will use the COCO dataset. COCO stands for Common Object in Context and is a dataset published by Microsoft that contains images in context to advance image recognition. COCO also has a bounding box for images needed when training an ML model to recognize objects. This saves a lot of time since finding and correcting the bounding box of objects can be quite time-consuming. Some of its features are:

- Object segmentation

- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints

(“COCO - Common Objects in Context,” n.d.)

2.3 Related work

DrNeil. 2022. Published an article featured on the Microsoft documentation site for using MR and Azure services to develop on HoloLens 1st Gen. The article goes through how to recognize custom content and its spatial positioning within an image using object detection. This is done using Azure custom vision and camera capture from Microsoft HoloLens 1.

They go through the following to make an MR application:

1. The user can gaze at an object that they have trained using the Azure Custom Vision Service, Object Detection.
2. The Tap gesture will be used by the user to capture an image of what they are looking at.
3. The image will be sent to the Azure Custom Vision Service by the app.
4. The Service will respond by displaying the result of the recognition as world-space text. This will be accomplished by using the Microsoft HoloLens spatial awareness to understand the world position of the recognized object and then providing the label text using the Tag associated with what is detected in the image.
   (DrNeil. 2022.)

Their solution uses HoloLens 1st Gen, Unity 2017.4 LTS, and some other tools.

Casano, Davide. 2022. Developed an application for HoloLens 2, which uses a customized object detector trained by Microsoft Azure. The primary aim of the thesis was to help and support people with the usage of a specific object watched or interacted with. Their

application uses a voice command, which, when pronounced, the HoloLens takes a picture and saves eye gaze coordinates. After receiving the response from Azure Custom Vision, a small audio and AR video guide that explains the usage of the object will appear.

Eckert, Blex, and Friedrich 2018. Wrote a journal article titled Object Detection Featuring 3D Audio Localization for Microsoft HoloLens A Deep Learning based Sensor Substitution Approach for the Blind. In which they propose a method for implementing a wearable, deep learning-backed object detection in the context of visual impairment or blindness. Their method made a prototype that uses Microsoft HoloLens using a near-real-time deep learning model named YOLOv2. The prototype can display and read out the names of the detected objects, which can be selected by voice command and then be used as a directional guide for the user using 3D audio feedback.

3.0 Proposed method

This chapter will go through one method on how to do object detection. This method will use a cloud service to do the object detection and then, with Unity as the main framework, make a UWP app that can be uploaded to the HoloLens using visual studio.

3.1 Method using cloud service for object detection

For this method, Microsoft Azure Custom Vision is used to train a model for object detection. This provides external GPU computational capabilities through a cloud system to actually to the detection and then send a response back. The images that the model's training images are taken from is the COCO dataset. The Custom Vision service will then be called using API from HoloLens to analyze the image and send a response.

3.1.1 Training the ML model

To upload images to Custom Vision can be done manually or with Custom Vision SDK. The SDK python API is selected to be used since it would be pretty time-consuming to upload manually.

The method used to upload images is the same as Casano, Davide. 2022. It is developed and described here with a few modifications. The following values are needed to connect to the API: Endpoint, training key, prediction key, and prediction resource id.

Since this project uses a student trial version of Custom Vision, the maximum number of images uploaded is capped at 100 000. When doing ML training, one should keep the number of images for each tag to roughly the same amount however an unbalanced dataset can be mitigated by weighting the categories. Because of this, the number of tags to be uploaded for each category is limited to 700 shown in Appendix A. This makes the distribution even, and the model will not be biased towards categories with more training tags and images.

The amount of time training for the model is somewhat limited since a student version of Azure Custom Vision is used however when trying to train for more hours, the results after training remained the same. Because of this, to get better accuracy, images are probably needed even more. After training, the results are precision, recall, and mAP. Precision is how likely the model is to be correct when it has detected a tag, recall is what percentage the model correctly found, and mAP mean average precision tells overall object detector performance across all tags.

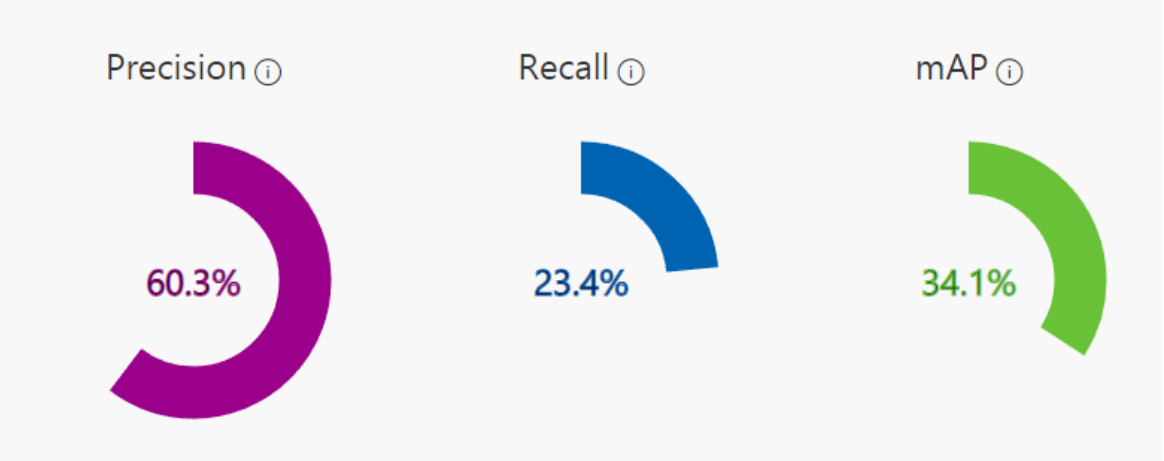

Figure 3.1: Results after training.

3.1.2 Application

The application was made in Unity 2020.3.30 using the latest Windows 10 SDK, Visual Studio 2022 as editor, and MRTK providing some components and features. When building the project from Unity, the platform is UWP, target device HoloLens with ARM64 architecture. A video of a demo where the application is used to do detection with HoloLens 2 can be found here.

After the application is started, the introduction text greets the user on how to use the program. For ease of use and to be user-friendly when capturing an image to be analyzed, voice commands can be used. In this case, the voice command is to *capture image*. Also, a voice command to reload the scene and remove all captured objects has been added with *remove objects*. Voice commands can easily be added by using the MRTK input speech system, then making a script shown in Appendix B to tell what the voice command should do.

After the user invokes the capture process, an image is taken and saved to a temporary folder. That image is then sent to Custom Vision for analysis. The image is sent using Custom vision API. A prediction key and prediction endpoint are needed to send the image for analysis. Custom Vision then sends a response in JSON format that has to be deserialized and is used by the SceneOrganiser script to find the real-world position the tag label should be placed. The script is shown in Appendix C.

The label text is first placed on a quad which is placed in front of the HoloLens position. This is done to find the direction the object is in. With spatial awareness enabled, the HoloLens will already have created how the world space looks with their sensors.

Then, a calculation is done to find what direction in the real world the object that was found is in. Since the response from Custom vision is in an (x, y) coordinate system going from 0 in the bottom left to 1 in the top right corner, and the object's location is calculated from the left and the top, this has to be transformed to Unity. The label will always be placed too much to the top right corner if no changes are made. In Unity, the center of the view is position 0 for x and y, with the bottom left being negative 0.5 and the top right being positive 0.5. To find the x position, it has to be moved to the left. This is done by taking x -

0.5. To find the y position, it has to be moved down from the top this is done by doing 0.5 - y. This is shown in *figure 3.2*.

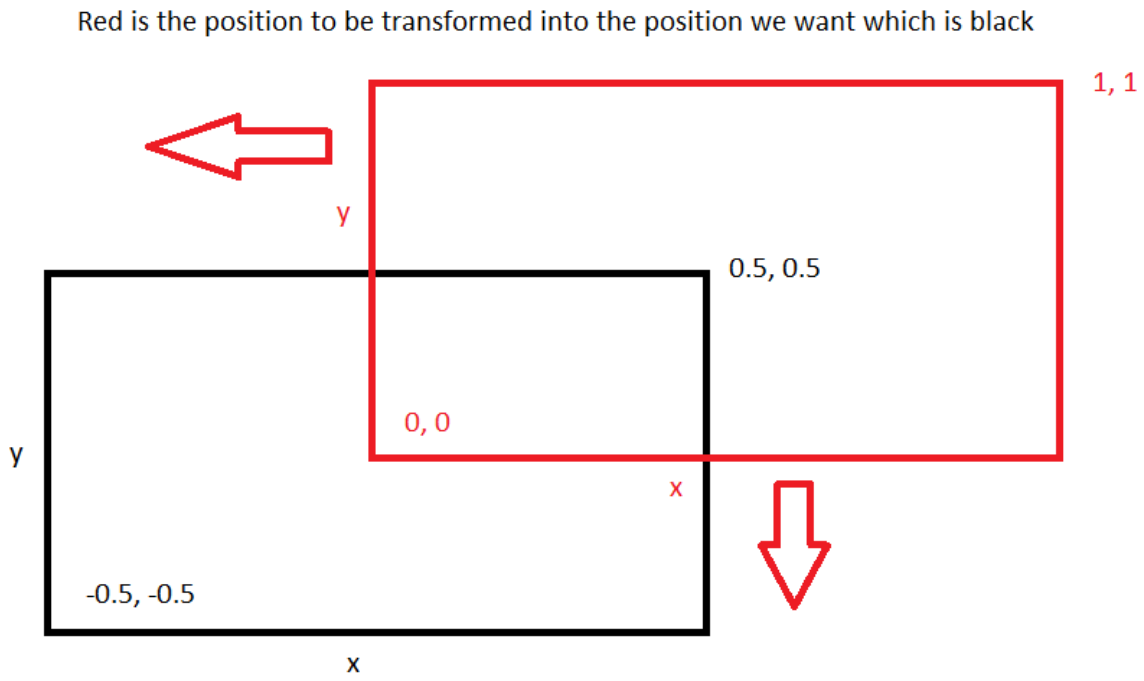

Figure 3.2: Calculates the position of the object that was found.

Suppose an object was detected with a probability higher than the minimum probability threshold, then a label will be placed on the surface the object was found. To find the real-world position, the direction of the object has to be found. A raycast is done to find the (x, y, z) axis of the point position of the object as shown in *figure 3.3*. This is done by taking the (x, y, z) coordinates of the HoloLens and (x, y, z) coordinates of the object found, placed on a quad invincible in the background. Since the quad is placed in the background of where the object was found, the position is the first point where something solid has been mapped from the spatial awareness in the object's direction. A component called lineRenderer adds a line from the camera position to where the object was detected to easier visualize where the raycast direction is.

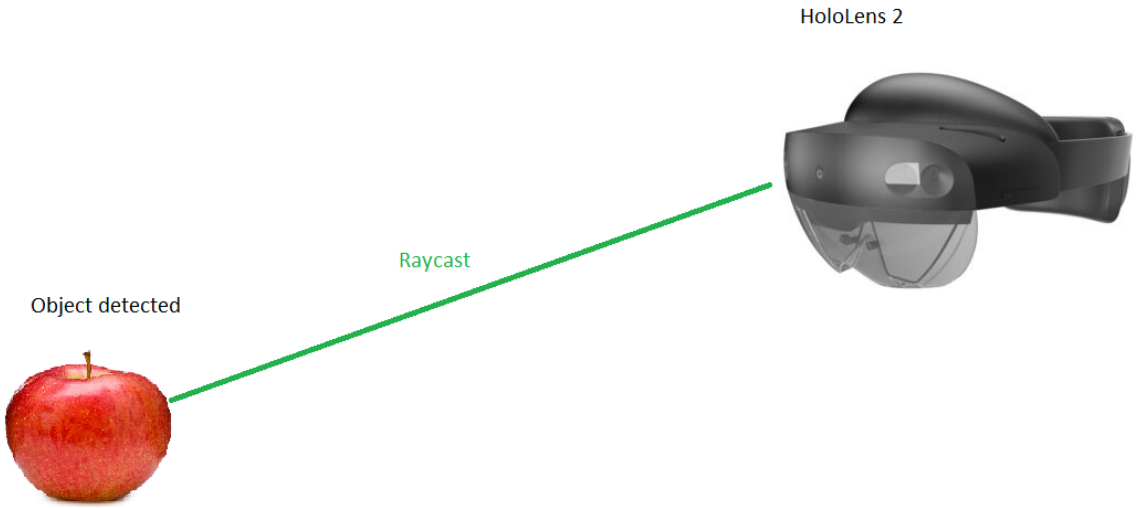

Figure 3.3: Raycast in the direction of the object detected.

4.0 Evaluation

This chapter will go through the evaluation of the application made by first going through the goal requirements and the validation of the application, which will also discuss how others can use this project and replicate or build on it. And then finally some limitations that come with this proposed method.

4.1 Goal requirements

The first requirement, as stated in the goal of this project, is to implement a way to do object detection of specific objects in real-time. In the case of this project, the objects chosen are an apple, banana, carrot, cup, fork, knife, computer mouse, and orange shown in *figure 4.1*.

Figure 4.1: Objects the ML model trained on.

The second requirement is to use the prediction from the ML model to find out the object's tag in the field of view and the object's bounding box, which is needed to place the object's label in the real-world using MR.

4.2 Validation

To validate the program, it was decided to look at all the objects in different settings and placements to see how correct the placement and labeling of the objects were. To determine what should be considered the wrong position the criteria were set that anytime the label is not placed directly on top of the object, or right under the object on the surface is considered wrong positioning. Also, the minimum threshold of the probability was set to 0.5. That way, no detection would be made anytime the probability was under 50%.

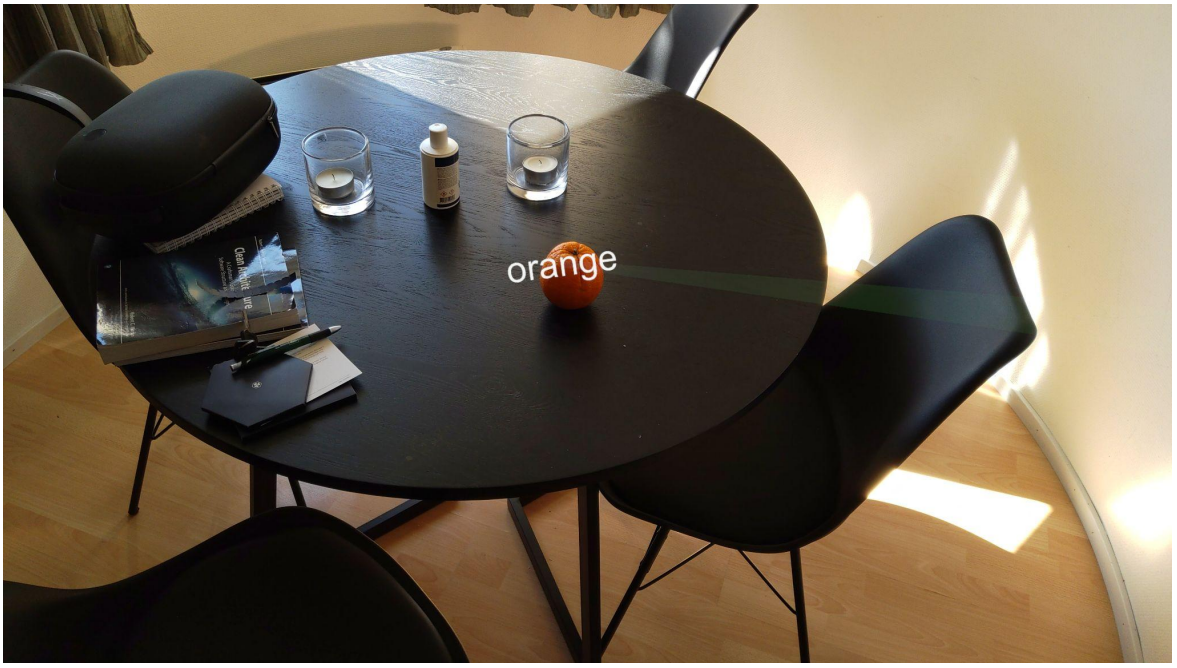

Figure 4.2: Example of an orange being detected.

The reason for the pick of the objects was first of all because these objects are in the COCO dataset which means there were a lot of images to upload to Custom Vision for training. The objects were four different kinds of fruit and four objects commonly found in most apartments.

Then ten images were taken of all the objects in different positions and contexts to see how many were correct, with the minimum threshold set to 0.5.

Table 4.1: Results show out of 10 trail detections for each object, how many were either correct label, correct position, incorrect label, incorrect position, or not detected.

	Correct label	Correct position	Incorrect label	Incorrect position	No detection
Apple	8	7	2	3	0
Banana	9	8	1	2	0
Carrot	7	6	3	4	0
Cup	7	6	2	3	1
Fork	5	5	3	3	2
Knife	5	5	2	2	3

Mouse	7	6	0	1	3
Orange	7	5	3	5	0

As can be seen, in general, more labels were correct than the positioning. Out of the 80 images taken, 55 had the correct label, 16 had an incorrect label, 48 had the correct positioning, 23 had incorrect positioning, and 9 had no detection. That means the correct label was 68.75 %, the correct position was 60 %, the incorrect label was 20%, the incorrect position was 28.75%, and no detection made was at 11.25%.

Generally, the fruits were detected more compared to the non-fruit items. One of the reasons for this could be because the fruits have a less complex and less varied shape and form. Therefore, they are easier to recognize. For example, a cup could have many different shapes and colors, while a banana usually will be either green or yellow. Because of this, more images are needed to train an ML model accurately on some objects compared to other objects.

When it comes to the position places, it was more incorrect compared to the label. This was usually because the HoloLens spatial awareness did not completely accurately detect the surface the object was on. In general, the further away from the object the user was standing the less likely the object was to be detected, and more likely to be incorrect.

Overall the trails went well, with more than 50% correctly labeled and positioned. When it comes to what could be done to make the program better, the most obvious thing is to use even more images for the training of the ML model. Seven hundred images is a lot, but when considering how many different contexts exist and how different the same object can look, many more images are needed for accurate training.

Since this project was made to be a proof-of-concept of a method to do object detection, the link to the Github of the project can be found here. The application has only been tested on the HoloLens 2. However, it is also possible to run the program through the Unity play mode. The Unity project will then detect and use a webcam connected to the computer if found. The text graphic will then be placed at the same distance always from the camera since this method does not have spatial awareness and cannot determine distance. Since Unity also is made to be cross-platform, it is possible to run the same solutions on other devices without needing to change too many of the settings. Note that in the

CustomVisionAnalyser script, a prediction key and prediction endpoint from Azure Custom Vision need to be added. Therefore an account is needed and a model needs to be trained by adding images of the objects to be detected.

4.3 Limitations

Some of the limitations that were noticed from the testing were that objects generally were too small to be registered by the HoloLens spatial awareness system. Because of this, the label usually was placed on the surface of which the object was placed, and not directly on the object. When objects were placed, for example on the edge of a table since the object is too small to be registered as a surface by the HoloLens. The raycast then goes "through" the object, registers the first surface, the wall or ground behind the object, and places the label there. If the object was placed on a glass table, the HoloLens would not register that the table is there and place the label on the ground, as shown in *figure 4.3*.

Figure 4.3: Example object label gets placed on the floor instead of the glass table, the numbers are the probability score of the class.

This program only places a label for the object with the highest probability of being correct. When there are multiple objects in the field of view, the object that was not intended to be

looked at, the non-targeted objects were sometimes registered instead. Other times objects such as tealight holders, as seen on the table in *figure 4.2,* were registered as cups instead of the object intended to be detected. Another limitation of this method is that the HoloLens must be connected to the internet through WIFI to use Microsoft Azure. Being required to be connected to WIFI limits the possible use cases to only places with stable WIFI connections. Using an external cloud service also introduces some latency while waiting for the response.

5.0 Discussion

For this project, a HoloLens 2 was chosen as the MR headset. The main reason for this is that it is more powerful than most other MR headsets currently on the market. Another reason is that there is much documentation on developing on HoloLens. One drawback with HoloLens is that it is quite expensive. Some alternatives are the following, magic leap one and raptor AR headsets.

For developing the application, Unity was chosen as the developing platform. As mentioned before, Unity is a cross-platform developing tool, which means the same Unity project can be used for many different devices and platforms. Microsoft also recommends using Unity, and they even created MRTK for Unity which is Microsoft-driven and open source, to help develop MR programs and applications.

When it comes to training the model and how to run the ML inference, the options stood between a cloud service like Azure and running the ML model directly on the HoloLens using GPU. It would be possible to create an application in Unity that can run inference on the HoloLens using Barracuda, a lightweight inference library, and other methods. As mentioned, when using a cloud service, there is a requirement that the MR headset has a stable internet connection. This would not be needed if running the model directly on the HoloLens. This was not seen as an issue for this thesis but could be something to keep in mind for others when doing similar projects. There is also a slight latency by using this method, but this issue was not seen as problematic again. One benefit of using a cloud service is that there is less computational demand for the MR headset. This could be important to keep in mind depending on the specifications of the MR headset. Using Azure

Custom Vision was chosen as this method seemed easier to implement, and the drawbacks were not seen as very impactful.

6.0 Conclusion and future outlook

The goal of this thesis was to use ML to recognize objects and use an MR headset to place the label of the detected object in real-world space, close to its actual location. Overall this goal was achieved for the objects trained using ML, a cloud system. Although the HoloLens does have good enough specifications to run an ML algorithm on its own, there are some benefits to using a cloud service like Custom Vision. Since the computational, power of the HoloLens is limited a GPU from an external cloud service makes sense to take some of the load away from the Hololens. More computing happening on the HoloLens also means the battery will drain faster and reduce the time between charging. There were some limitations with the proposed method, as mentioned in 4.3, but otherwise, the method achieved the proposed goal. Although the proposed method has only been tested on HoloLens 2 and through Unity play mode, since Unity is made to be cross-platform, it should be possible to use it for other devices.
During this thesis, the following was implemented successfully, created a method, created an application, and was able to identify objects in near-real-time and place the labels of these objects in real-world space. The HoloLens can create a sound world understanding through its spatial awareness capabilities which can potentially be used to create a great variety of use cases both for object detection Applications but also other MR projects as well

6.1 Future outlook

Possible extensions to the application could be to place labels for each unique object that was registered over the set threshold to be able to detect and use MR for multiple objects at a time. This would help solve problems where the wrong object was detected because another object was in the field of view. It could also be possible to do object detection based on head and eye gaze. The label then gets placed based on that instead of object direction. Other methods could use ML models made from scratch to do specific detection for the use case needed. This method could also do ML inference without being connected to WIFI,

which expands the possible use cases for where the HoloLens can be used. As shown in this thesis, the HoloLens, together with Unity, can create engaging MR experiences with a wide variety of applications.

The future of MR is looking bright as the technology evolves, with headsets becoming smaller and less cumbersome at the same time as the technology is advancing, and with companies such as Meta embracing and calling the metaverse the next digital frontier. Less time will be spent on computer and phone screens as these will seem more limiting and not as immersive as MR headsets become more widely available.

Appendix A

```
#Check how many images there are
for x in range(len(category_ids)):

    # actual progress percentage
    progress -= 1
    actual_progress = ""#str(100 - (progress * 100 / total_length))
    print("[" + actual_progress + "%] Adding class:", sel_categories[x])

    unique = []
    image_ids = []
    for j in range(n_entries):
        category_id = data["annotations"][j]["category_id"]

        if category_id == category_ids[x]:
            image_id = data["annotations"][j]["image_id"]
            image_ids.append(image_id)
            unique = np.unique(image_ids)
        if len(unique) >= 700:
            break

    ###

    image_ids = np.unique(image_ids)
    image_indexes = [j for j, y in enumerate(data["images"]) if y["id"] in image_ids]
    print("Number of images", len(image_indexes))
```

Listing 1: Limit the number of tags for each category.

Appendix B

```
    public void OnSpeechKeywordRecognized(SpeechEventData eventData)
    {
        Debug.Log("OnSpeechKeywordRecognized");

        switch (eventData.Command.Keyword)

        {
            case "Capture Image":
                ImageCapture.Instance.Invoke("ExecuteImageCaptureAndAnalysis", 0);
                Debug.Log("Capture Image");

                break;

            case "Remove Objects":
                SceneOrganiser.Instance.Invoke("reload", 0);
                Debug.Log("Remove Objects");

                break;
```

```
        }
    }
```

Listing 2: Speech commands script.

Appendix C

```
    public void FinaliseLabel(AnalysisRootObject analysisObject)
    {
        if (analysisObject.predictions != null)
        {
            lastLabelPlacedText = lastLabelPlaced.GetComponent<TextMesh>();
            // Sort the predictions to locate the highest one
            List<Prediction> sortedPredictions = new List<Prediction>();
            sortedPredictions = analysisObject.predictions.OrderBy(p =>
            p.probability).ToList();
            Prediction bestPrediction = new Prediction();
            bestPrediction = sortedPredictions[sortedPredictions.Count - 1];

            if (bestPrediction.probability > probabilityThreshold)
            {
                quadRenderer = quad.GetComponent<Renderer>();
                Bounds quadBounds = quadRenderer.bounds;

                // Position the label as close as possible to the Bounding Box of the
prediction
                // At this point it will not consider depth

                lastLabelPlaced.transform.parent = quad.transform;
                lastLabelPlaced.transform.localPosition =
                    CalculateBoundingBoxPosition(quadBounds, bestPrediction.boundingBox);

                lastLabelPlacedText.text = $"{bestPrediction.tagName}";



                RaycastHit objHitInfo;
                Debug.Log("Repositioning Label");
                Vector3 headPosition = Camera.main.transform.position;
                //Direction of the object detected
                Vector3 dir = lastLabelPlaced.transform.position -
                    Camera.main.transform.position;

                //Visualize ray to see where raycast went
                laserline = GetComponent<LineRenderer>();
                laserline.SetPosition(0, headPosition);

                //When Raycast hits the mesh added from spatial awareness assumes thats where
the object is
                if (Physics.Raycast
                    (headPosition, dir, out objHitInfo, 30.0f, Physics.DefaultRaycastLayers))
                {
```

```
                lastLabelPlaced.position = objHitInfo.point;
                laserline.SetPosition(1, objHitInfo.point);
            }
            StartCoroutine(FadeLineRenderer());
        }
    }

    ImageCapture.Instance.ResetImageCapture();
}
```

Listing 3: Finalises the position of the label.

Bibliography

Jordan, Michael I., and Tom M. Mitchell. "Machine learning: Trends, perspectives, and prospects." Science 349, no. 6245 (2015): 255-260. https://doi.org/10.1126/science.aac4520.

Jason Brownlee. 2019. “A Gentle Introduction to Object Recognition with Deep Learning.” Machine Learning Mastery. July 5, 2019. https://machinelearningmastery.com/object-recognition-with-deep-learning/.

Scooley. 2020. “HoloLens 2 Hardware.” Microsoft.com. October 20, 2020. https://docs.microsoft.com/en-us/hololens/hololens2-hardware.

Microsoft. 2019. “HoloLens 2—Overview, Features, and Specs | Microsoft HoloLens.” Microsoft.com. 2019. https://www.microsoft.com/en-us/hololens/hardware.

“7 Things about Microsoft Hololens.” n.d. https://www.gvsu.edu/cms4/asset/7E70FBB5-0BBC-EF4C-A56CBB9121AECA7F/7_things_about_microsoft_hololens.pdf.

Mokhtari, Nour Islam. 2021. “Deep Learning for Object Detection : Beginners Friendly Guide.” Medium. November 28, 2021. https://towardsdatascience.com/deep-learning-for-object-detection-beginners-friendly-guide-ae1180b34042.

Caseymeekhof. 2022. “Point and Commit with Hands - Mixed Reality.” Docs.microsoft.com. Accessed May 7, 2022. https://docs.microsoft.com/en-us/windows/mixed-reality/design/point-and-commit.

Nitinme. 2022. “What Are Azure Cognitive Services? - Azure Cognitive Services.” Docs.microsoft.com. https://docs.microsoft.com/en-us/azure/cognitive-services/what-are-cognitive-services.

PatrickFarley. n.d. “What Is Custom Vision? - Azure Cognitive Services.” Docs.microsoft.com. Accessed April 1, 2022. https://docs.microsoft.com/en-us/azure/cognitive-services/custom-vision-service/overview.

BrandonBray. 2022. “What Is Mixed Reality? - Mixed Reality.” Docs.microsoft.com. https://docs.microsoft.com/en-us/windows/mixed-reality/discover/mixed-reality#:~:text=Mixed%20Reality%20is%20a%20blend.

polar-kev. 2022. “MRTK-Unity Developer Documentation - Mixed Reality Toolkit.” Docs.microsoft.com. https://docs.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/?view=mrtkunity-2021-05.

DrNeil. 2022. “HoloLens (1st Gen) and Azure 310 - Object Detection - Mixed Reality.” Docs.microsoft.com. https://docs.microsoft.com/en-us/windows/mixed-reality/develop/unity/tutorials/mr-azure-310.

Casano, Davide. 2022. “HoloHelp: HoloLens Object Detection for a Guided Interaction.” GitHub. February 21, 2022. https://github.com/davide-cas/HoloHelp.

PatrickFarley. 2022. “Quickstart: Object Detection with Custom Vision Client Library - Azure Cognitive Services.” Docs.microsoft.com. Accessed April 13, 2022. https://docs.microsoft.com/en-us/azure/cognitive-services/custom-vision-service/quickstarts/object-detection?tabs=visual-studio&pivots=programming-language-python.

Milgram, Paul, and Fumio Kishino. "A taxonomy of mixed reality visual displays." IEICE TRANSACTIONS on Information and Systems 77, no. 12 (1994): 1321-1329.

https://www.researchgate.net/profile/Paul-Milgram/publication/231514051_A_Taxonomy_of_Mixed_Reality_Visual_Displays/links/02e7e52ade5e1713ea000000/A-Taxonomy-of-Mixed-Reality-Visual-Displays.pdf

Eckert, Martin, Matthias Blex, and Christoph M. Friedrich. "Object detection featuring 3D audio localization for Microsoft HoloLens." In Proc. 11th Int. Joint Conf. on Biomedical Engineering Systems and Technologies, vol. 5, pp. 555-561. 2018. https://pdfs.semanticscholar.org/5081/19a50e3d4e8b7116c1b56a002de492b2270b.pdf

Mystakidis, Stylianos. "Metaverse." Encyclopedia 2, no. 1 (2022): 486-497. https://doi.org/10.3390/encyclopedia2010031.

“COCO - Common Objects in Context.” n.d. Cocodataset.org. https://cocodataset.org/#home.