UiT
THE ARCTIC UNIVERSITY OF NORWAY

Faculty of Engineering Science and Technology

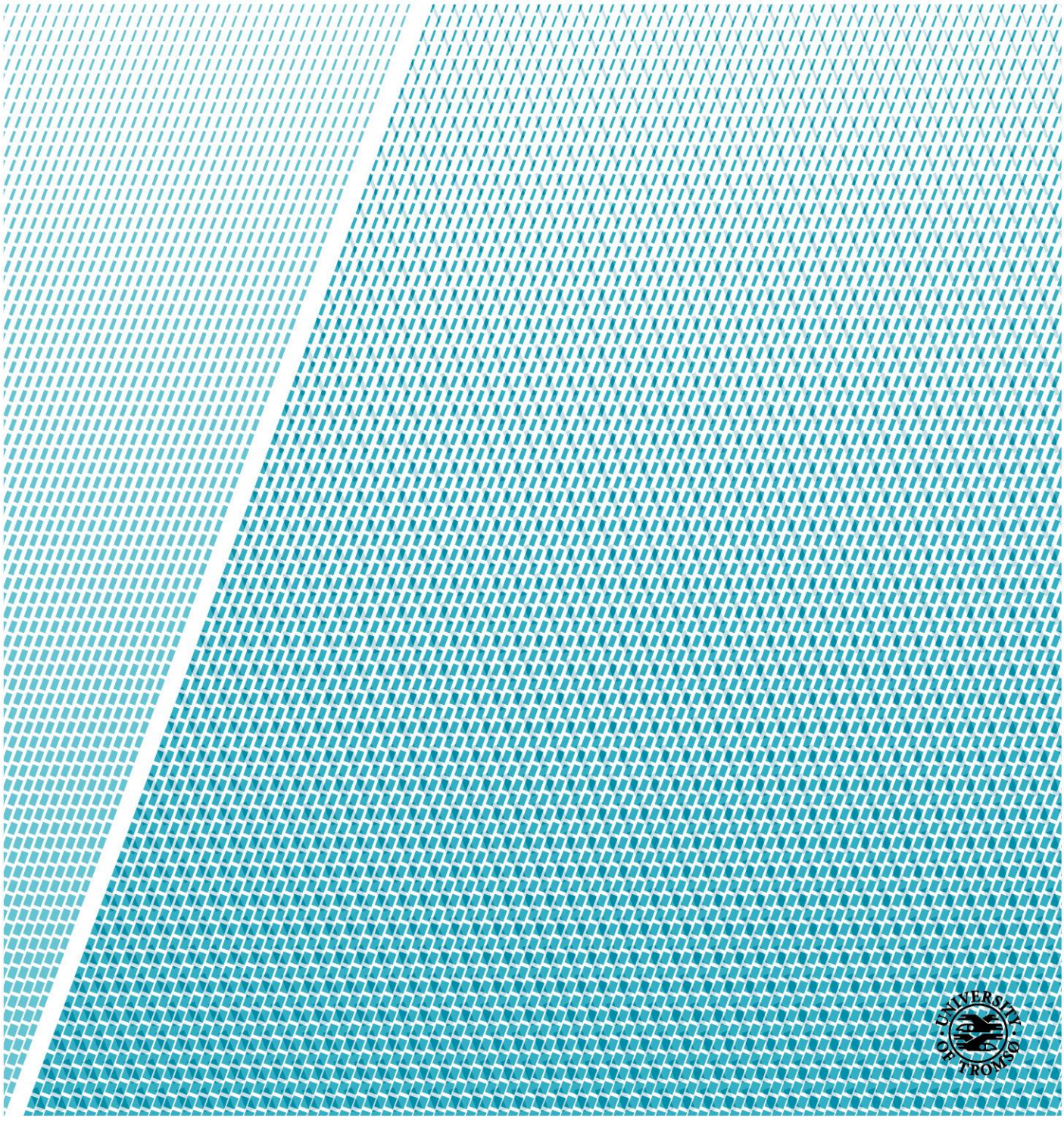
Department of Computer Science and Computational Engineering

Augmented Reality Application for Training in Maritime Operations

A Proof of Concept AR Application Developed for Microsoft® HoloLens

Hui Xue

Master in Computer Science, June 2017



Preface

This thesis is submitted in fulfilment of the requirement for the Master Degree at UiT The Arctic University of Norway. The work described in this thesis was carried out in the Department of Computer Science and Computational Engineering in 2017. It is the original and independent work of author except where specially acknowledged in the text. Neither the present dissertation nor any part thereof has been previously submitted at any other university. This dissertation contains approximately 14220 words, 84 figures and 2 tables.

Hui Xue (Helene)
Department of Computer Science and Computational Engineering
UiT The Arctic University of Norway
June 2017

Acknowledgements

The thesis consumed a lot of my time and required dedication. It would not have been possible for me to complete if I did not have a support of many individuals and organizations. Therefore, I would like to extend my sincere gratitude to all of them. First and foremost, I would like to express my sincere gratitude to my supervisors: Prof. Børre Bang and Dr. Puneet Sharma. I received useful comments and engagement from them throughout the learning process of my master thesis. Without their assistance and dedicated involvement in every step, this thesis would have never been accomplished. I would also like to show gratitude to Long Qian from Johns Hopkins University, who devoted his time and knowledge for wonderful discussions via emails and messages. Furthermore, I would like to thank Bjørn-Morten Batalden from Department of Engineering and Safety in UiT for sharing the maritime operation document with me. Without his support, it would not be possible for me to understand the ship bridge. Getting through my dissertation required more than academic support, and I thank my best friends Linqian Li and Zhiyin Liu for listening and, at times, having to tolerate me over the past five months. I would like to thank my part-time job fellows in Sushi Point Narvik for giving me convenience during my leave. Most importantly, none of this could have happened without my family. My parents offered their encouragement through phone calls every week. I would like to thank them for supporting me spiritually and advising me on the matters of life in general.

Hui Xue (Helene)
Tromsø, June 2017

Abstract

Augmented reality (AR) is an advanced technology that integrates augmentations with the real world. This technology has been used to provide training and education along with other purposes. This work has been focused on enriching the learning experience of the maritime trainee by applying AR technology.

In this work, a proof of concept AR application (App) is developed for the training of the maritime students. The App was designed to introduce the selected stations and panel in the Kongsberg Ship Bridge Simulator in Department of Engineering and Safety, UiT The Arctic University of Norway. The App is designed with four main options namely: Stations, Panel, Help and Quit. Microsoft® HoloLens has been chosen as a wearable AR device for the execution of the App. The primary reason for selecting Microsoft® HoloLens was its standalone ability.

In this App design, marker-based tracking and markerless-based tracking were compared. The marker-based tracking was found to be appropriate for the maritime training. The main reason is that marker-based tracking allows to adapt to the different Ship Bridges without much of a hassle. Within marker-based tracking method, various types of markers were considered. After thoughtful consideration, AR image markers and 2D-barcode markers were chosen. AR image markers are effective within larger range (up to 1.5m) hence useful to highlight the stations on the Ship Bridge. However, 2D-barcode markers are easy to detect (consume less processing power and allow multiple markers to be detected at the same time) and hence useful on the control panels. The camera on the HoloLens has to be calibrated for the accurate tracking of the markers.

The App was designed in Unity 5.5.1f1 Personal version, Visual Studio 2015 Update 3 and Windows 10 environment. The scripts were written in C# Programming Language and libraries from ARToolKit SDK, OpenCV, HoloToolKit were incorporated.

Developed App is limited to operate only with the Microsoft® HoloLens. Workings of the App can be viewed here: https://youtu.be/zzJ6mEH91F4.

Keywords: *Application (App), ARToolKit, Augmented Reality (AR), C#, Detection, Maritime Training, Marker, Microsoft® HoloLens, Ship Bridge, Tracking, Unity, Visual Studio, Wearable*

"Augmented Reality Will Be As Big As the iPhone."

Tim Cook, CEO Apple Inc.

Table of contents

List of figures

List of tables

List of Abbreviations

The list describes several symbols that will be later used in the dissertation.

API	application platform interface
App	application
AR	augmented reality
AV	augmented virtuality
CLR	Common Language Runtime
CT	Computed Tomography scan
DLL	dynamic link library
EDC	error detection and correction
FoV	field of view
GPS	Global Positioning System
HMD	head-mounted display
HPU	Holographic Processing Unit
HUD	heads-up display
MRI	Magnetic Resonance Imaging
OHMD	optical see-through head-mounted display
OpenCV	Open Computer Vision
OpenGL	Open Graphic Library

OS	operating system
RGB	red green blue
SDK	software development kit
SSAT	Specialist Schools and Academies Trust
UWP	Universal Windows Platform

Chapter 1

Introduction

Augmented Reality (AR) is a system that overlays virtual objects onto an image of the real world. AR elements are augmented by computer-generated sensory input such as sound, video, graphics, accelerometer or GPS data [1]. A key measure of AR systems is how realistically they integrate augmentations with the real world. Recent advances in AR technology have enabled it to use to provide training and education along with other purposes. The AR system can provide the necessary insight or information to improve a trainee's performance.

In maritime operations, experienced operators from captain to crew play a crucial role. There is a continuous flow of information from one to another. This information plays the key role in making correct decisions and performing successful maritime operations. AR technology can enrich the learning experience of the maritime trainee and allows sharing the expertise (knowledge and experience) of an expert to a novice in a personalized manner.

This thesis investigates the application of AR technology in the maritime training. The objective was to develop an introduction of certain stations and panels of a ship bridge using Microsoft® HoloLens. The app was developed in Unity 3D project. The scripts were written in C# and deployed in Microsoft® HoloLens.

1.1 Augmented Reality (AR)

In 1994, Paul Milgram introduced the reality-virtual continuum [2] with the scale ranges from a real word on one end to an entirely virtual environment on the other end. Since then various efforts has been made in the field, but from the applied side, augmented reality (AR) is still relatively new technology. Figure 1.1 shows that AR is a form of mix reality and combines real world and virtual information. As people might be familiar with playing a game on a Nintendo Wii or watching an IMAX movie that was defined as augmented

virtuality (AV), AR can be described as the inverse of AV, where virtual objects are layered over the real environments [3].

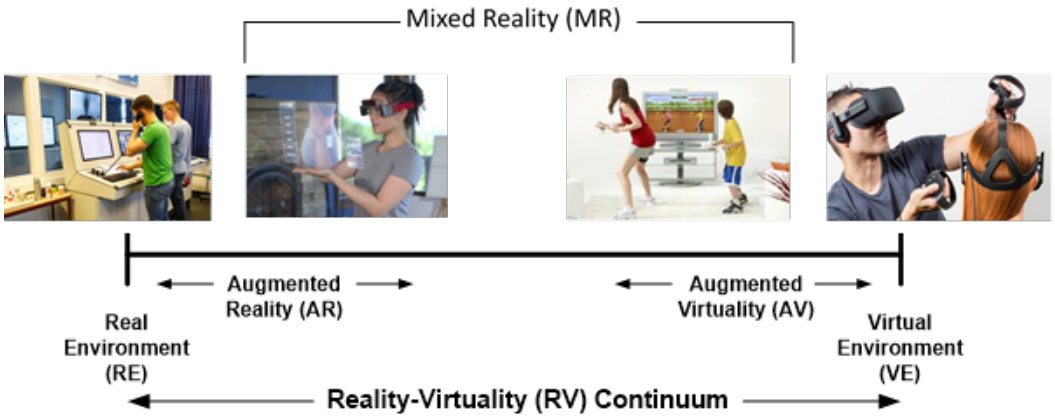


Fig. 1.1 Reality-virtuality continuum [2]

In AR, users interact digitally with the information surrounding the real world. By adding computer vision and object recognition, users can see the information about the environment. In AR virtual objects are superimposed upon the real world around them. The information conveyed by the virtual objects help the users perform real-world tasks [4]. Nowadays, AR technology has been widely used in medical visualization, maintenance, and repairs, training, and targeting. AR is supported by hardware components such as processor, display, sensors and input devices. It is experienced with a variety of devices such as smartphones, PC with a webcam, laptop and wearable devices. In this study, for the reason of giving better user experience and making it more user-friendly, the wearable AR device is used. The wearable AR device is the best choice for the teaching application since hands are free to interact. Therefore, wearable AR devices will be focused in this thesis.

1.1.1 Wearable AR Devices

The basic idea of AR is to overlay the virtual objects over a real-world environment in a real-time. AR displays are rendered by using various technologies such as optical projection systems, monitors, smartphones and optical see-through head-mounted display (OHMD). For the reason that smartphone or handheld devices have small screens to superimpose information, OHMD provides users with more convenient, expansive views of the world around them. An OHMD is a wearable device that combines sensors for tracking and registration objects between the superimposed data and the real world. There are many

wearable AR devices currently available in the market, the most popular ones are listed as follows [5] (Figure 1.2):

- Microsoft® HoloLens
  Microsoft® HoloLens is a pair of mixed reality smart glasses with high-definition 3D optical head-mounted display and spatial sound which was introduced in 2016. It has Windows 10 operating system (OS) with AR unit using the Windows Holographic platform. Microsoft® HoloLens can see, map, and understand the physical places, spaces, and things around the users based on its advanced sensors [6].

- Google Glass
  Google Glass became available to the public market in 2014. It is developed by X (previously Google X) [7]. It displays information in a smartphone-like hands-free format [8]. It works with smartphones as well as a variety of Android-friendly third-party apps. It can take pictures and shoot videos of what user is looking at through glasses.

- Sony SmartEyeGlass
  Sony SmartEyeGlass was first introduced in 2014, and it serves as a heads-up display (HUD)for the Android devices. It includes different types of sensors such as an accelerometer, gyroscope, electronic compass, brightness sensor, microphone and noise suppression sub microphone [9]. The text appears over the lenses which provide the user with the information such as directions, location, and real-time voice translations.

- Epson Moverio BT-300
  Epson Moverio BT-300 smart glasses was introduced in 2016, and it has been improved a lot from the previous versions with a lighter and more appealing pair of glasses [10]. It can easily connect to a wide range of devices to receive content and data at high speed. It also can be interacted via a special cable connected control box that comes with a large touch panel. BT-300 now uses an Intel Atom 5, 1.44 GHZ Quad core chip and has an Android 5.1 OS which widely expands the complexity of apps that can be written for the glasses [11].

To find best AR technology in training, advantages and disadvantages of all above discussed wearable devices are compared in Table 1.1. It is shown that only Microsoft® HoloLens is standalone and it completely frees the hand from the device.

Considering Microsoft® HoloLens is the only standalone device. Therefore, it is a clear choice for development of AR based training in maritime operations. Therefore, in the thesis,

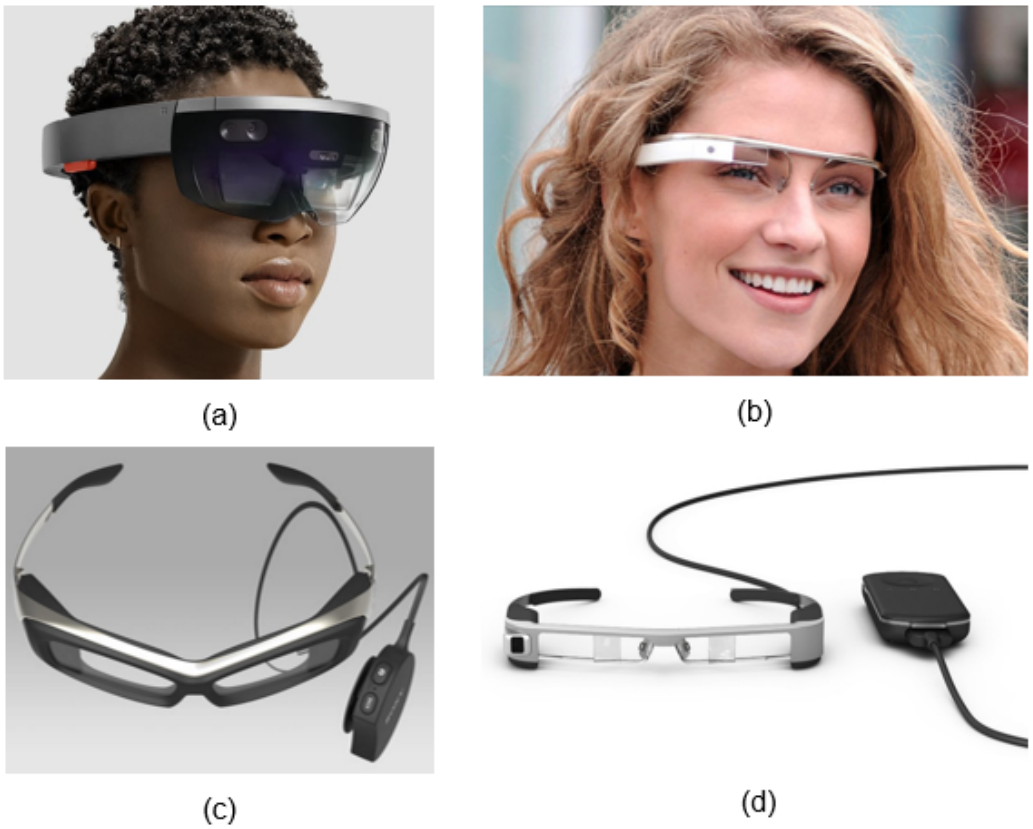


Fig. 1.2 Variable wearable AR devices(a) Microsoft® HoloLens [6]; (b) Google Glass [7]; (c) Sony SmartEyeGlass [9]; (d) Epson Moverio BT-300 [11]

Devices	*Release Date*	*Operating system*	*CPU*	*Display*	*Controller input*	*Weight*	*Stand alone*
Microsoft® HoloLens	2016	Windows Mixed Reality	Intel 32-bit (1GHz)	2.3 megapixel widescreen stereoscopic head-mounted display	Gestural commands via sensors and HPU	579g	Yes
Google Glass	2014	Glass OS	OMAP 4430	Prism projector, 640×360 pixels	Touchpad, MyGlass phone mobile app	36g	No
Sony SmartEyeGlass	2015	Android	None	Monochrome (green)	Android device	77g	No
Epson Moverio BT-300	2016	Android 5.0	1.44GHz Quad Core	OLED display	BT-300FPV controller	69g	No

Table 1.1 Comparison between the four different Optical see-through head-mounted display devices [12, 13]

I would use this device to develop a working application to demonstrate the idea of maritime training.

1.1.2 Input devices for wearable AR devices

Traditional input devices such as keyboards and mouse do not support the wearable AR devices. The input devices such as speech recognition, gesture recognition, eye tracking and buttons are widely used in wearable AR devices because of its mobility and hands-free use. For example, gesture recognition captures users' hand gestures by gesture camera [14]. Figure 1.3 shows the example of gesture command in Microsoft® HoloLens. In this case, when the user performs the gesture of Air Tap on the object where it was gazed, a new object will be generated.

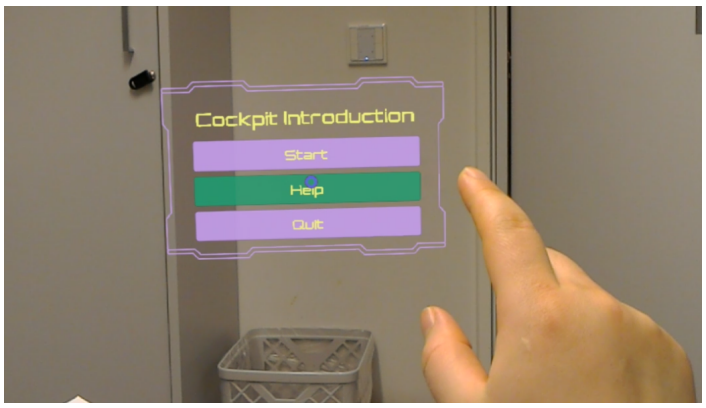


Fig. 1.3 An example of gesture recognition by Microsoft® HoloLens

1.2 AR in Maritime Operations Training

Smart ships and advances in marine technology require mariners to be well trained. Training mariners in navigation are very expensive since it requires an extensive number of hours in the sea. Nowadays, it is possible to provide real-time sea experience with additional safety in ship simulators (for example Kongsberg Ship Simulators, UiT The Arctic University of Norway). This training can be made even more interactive and engage with the AR technology. The combination of AR innovation with educational contents opens up a whole realm of applications, especially in self-education and skill developments. For example, with AR device, an inexperienced crew member can gain professional experience and training in his own time just by following the information in the AR application.

1.3 Problem Overview

The objective of the thesis is to develop the application (app) in Microsoft® HoloLens and conduct suitability trials. Getting the trainees to know the station of the bridge is the main idea of the app. In this app, there are four options in the main menu: Stations, Panels, Help, and Quit. Stations option detects markers for Main Steering Station and Dynamic Position Operator Station. Panels introduces available controls on the Main Steering Station. These are Throttle Control, Azimuth Control, Thruster, and Rudder. Help option explains the user the available options and voice command in this application. Quit option exits the application. To guide the trainees to find correct position in the bridge, the station needs to be detected and recognized. In this study, I have used the marker detection technology in Microsoft® HoloLens. The information will be shown in the Microsoft® HoloLens as the visual object such as text or 3D models accordingly. In each of scenario, different types of markers such as traditional AR marker and 3×3 2D-barcode markers were used for different requirement. There is a live video in each scenario helps the user to identify the markers. Voice command "Back" will take the user from the current option to the main menu. Air Tap different buttons will give corresponding action.

The App developed in Unity 3D and the code is written in C#. Some of the 3D model such as right-hand pointing model was designed in Autodesk® Maya. The markers were designed in Adobe® Illustrator CC. The app was designed for the Ship Bridge Simulators in UiT- The Arctic University of Norway as shown in Figure 1.4 and Figure 1.5.

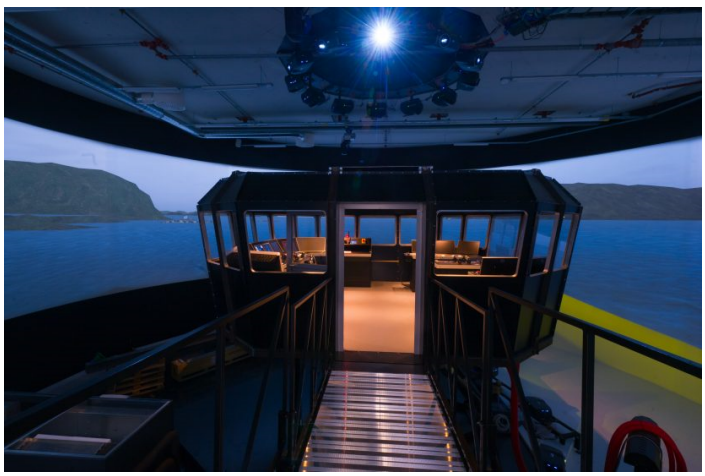


Fig. 1.4 The Ship Bridge simulator in UiT [15]

Fig. 1.5 Inside view of the Ship Bridge simulator in UiT [15]

1.4 Thesis Outline

This report is divided into 5 Chapters. The contents of the rest chapters are described as follows,

- Chapter 2 gives the literature review on the augmented reality. The focus is gathering necessary information for making a useful application on the maritime operation. The chapter provides variety method how the AR technology approached to the different training application.
- Chapter 3 presents the methodology. There are three main sections: Microsoft HoloLens, Application (App) development and Application (App) design.
- Chapter 4 presents the results and discussion.
- Chapter 5 gives the conclusion and future research directions in relation to this project.

In addition, the related material for this project is provided in appendix A. Demo video of the working of the application can be seen from the links: https://youtu.be/zzJ6mEH91F4.

A list of references is provided at the end of the report.

Chapter 2

Literature Review

Azuma [4] defined augmented reality (AR) as a system that allows the user to see the real world which has virtual objects superimposed upon or composed. The AR system has three characteristics: combines with real and virtual objects; interactive in real time, and registered in a three-dimensional (3D) [16].

The first appearance of Augmented Reality (AR) dates back to the 1950s [17]. By the time of the 1990s, AR emerged as a named field in computer science. However, it was not gained a lot of commercial and research interest until the mobile devices gained possibility to support AR applications [18]. With the new approaches of augmentations, it opened AR technology to a huge users group [19]. Therefore, AR applications have been explored in many areas such as education, medical, maintenance, repair, military, entertainment and maritime applications. Following are the case studies of AR applications.

2.1 AR in Education

AR technology becomes an interesting topic in training and education [20]. Researchers found out that combining real and virtual objects in 3D are helpful in reducing cognitive load [18]. The displayed virtual objects are not directly detectable by users own senses. AR training applications can enhance the user's perception and interaction with the real world. Research indicates that the AR applications as an educational tool can improve students' performance and their analytical skills. AR applications also can increase students motivational and engagement levels [21]. Therefore, AR has been applied in a variety of subjects.

- AR in Astronomy

Figure 2.1 shows that a classroom is augmented with the planet. Students can learn about the relationship between the planets by playing an AR application. Students will feel more involved and get more motivation about learning astronomy.

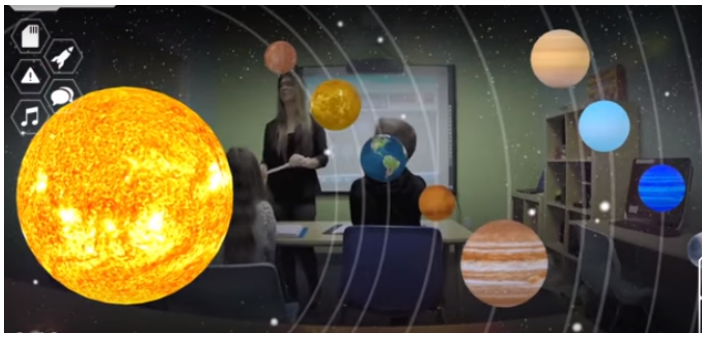


Fig. 2.1 An astronomy class [22]

- AR in Chemistry

AR application in chemistry teaching can demonstrate atomic and molecular structures to the students. Figure 2.2 shows that teacher holds the target picture and the molecular structure can be viewed from the screen.

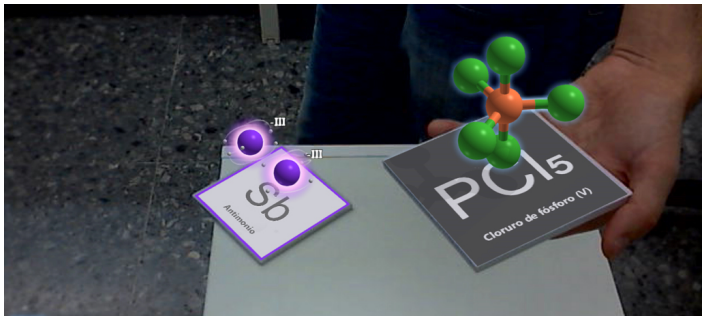


Fig. 2.2 AR is used in a Chemistry lesson [23].

- AR in Biology

AR application can help students to study the anatomy and structure of the body. The Specialist Schools and Academies Trust (SSAT) demonstrated that teachers could use AR technology to show the organs composition and appearance. Student will learn how the organs look by watching 3D computer-generated models in the classrooms. Moreover, students will be able to study humans' organs independently with AR markers and information. Figure 2.3 shows the example of AR application (developed by SSAT) allows the user to see the internal organs through a webcam connected to a computer [24].

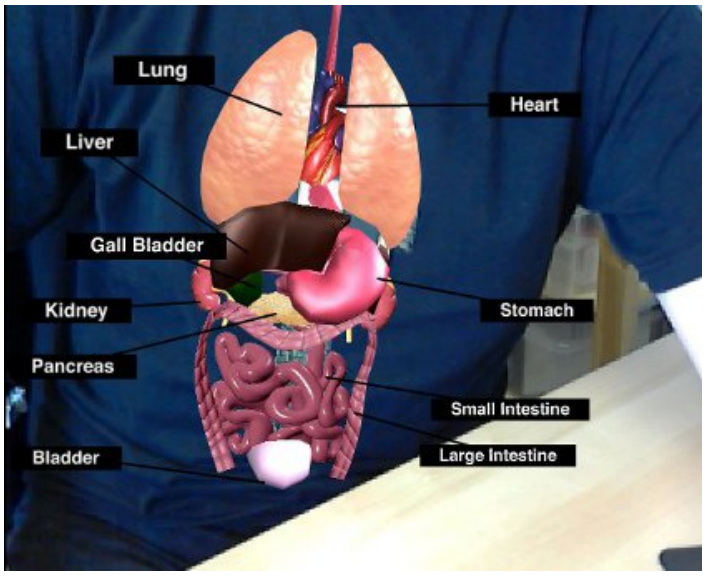


Fig. 2.3 AR application for studying humans' organs [24]

2.2 AR in Medical

Doctors can use AR application as a visualization and training aid for surgery [4]. The 3D data collected from the patient by Magnetic Resonance Imaging (MRI), Computed Tomography scans (CT), or ultrasound imaging can be rendered and overlapped on the patient in real time and real world. This can help the doctor to see an internal view without giving the patient an incision.

There are many research projects have been done to explore AR applications in the medical area. For example, a research group from UNC Chapel Hill has conducted trials to view the fetus from a see-through HMD [4]. These trials overlap the scan results over the womb of a pregnant woman (Figure 2.4). Another research group from NARVIS project has integrated an HMD-based (head-mounted display) AR system into the operation room for 3D in situ visualizations of CT images (see Figure 2.5). The system provides assistance for spinal surgery procedure.

2.3 AR in Military

In combat, AR can serve as a networked communication system that renders useful battlefield data onto a soldier's goggles in the real time. With AR system, commanders can send maps and other information directly to the soldier's field of vision. From the soldier's viewpoint, people and various objects can be marked with special indicators to warn of potential dangers.

Fig. 2.4 Virtual fetus inside the womb of the pregnant patient [25]

Fig. 2.5 Surgeons can detect some features of the face that they cannot see in MRI or CT scans [26].

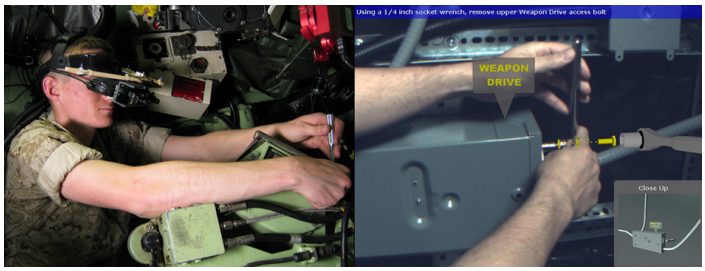
Virtual maps and 360° view camera imaging can also be rendered to aid a soldier's navigation and battlefield perspective. In addition, this information can be transmitted to military leaders at a remote command centre. The AR application could provide troops with vital information about their surroundings (see Figure 2.6).

Fig. 2.6 AR system can overlay key information such as the location of enemies, satellite footage and mission objectives [27].

2.4 AR for Maintenance, Repair

AR devices can play a role of tutor. The use of AR to aid in the execution of procedural tasks are explored in maintenance and repair area. In AR maintenance applications, the virtualization of the user and maintenance environment allows off-site collaborators to monitor and assist with repairs. Additionally, the integration of real-world knowledge databases with detailed 3D models provide opportunities to use the system as a maintenance simulator\training tool. Moreover, the research in this area has an objective of improving the productivity, accuracy, and safety of personnel. One of the research found out that by using AR applications in maintenance, trainees assembling airplane wing can reduce errors upto 90%. This was in comparison to the trainees who were given instructions through the desktop computer [28]. The result also found out that with the AR application, the assembly time was reduced by about 35% [28].

There are are a variety of AR applications designed for maintenance and repair. Figure 2.7 shows a mechanic is doing a maintenance task wearing a tracked head-mounted display. Through the head-mounted display, the information is provided to assist the mechanic [29].



(a) (b)

Fig. 2.7 (a) A mechanic wearing a tracked head-mounted display performs a maintenance task inside an LAV-25A1 armoured personnel carrier. (b) AR information through the head-mounted display to assist the mechanic [29].

Figure 2.8 shows a user is learning the procedure of car engine maintenance by using AR application in a smartphone. Figure 2.9 shows an AR application of a plant Maintenance by using HoloLens.

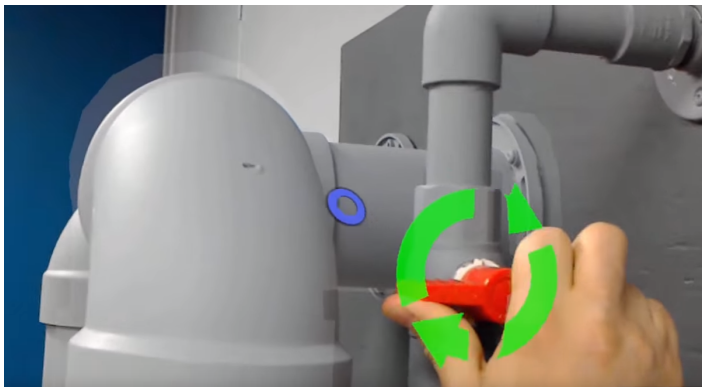
Fig. 2.8 Car maintenance with the help of AR application using a smartphone [30].

In above applications, it can be observed that a tracking system has been used with marker detection or object detection. The AR program draws graphics on the top of a particular object to present important information.

2.5 AR in Entertainment

AR has become common in entertainment. One of the reason is the popularity of smartphones supporting AR applications. There are many AR application developed for entertainment purpose. For example, with the help of AR application, tourist can get the information easily about the tourist attractions. Figure 2.10 shows a tourist using an AR application to check the

(a)



(b)

Fig. 2.9 AR application of a plant Maintenance by using HoloLens (a) Spectator view (b) View from the operator through HoloLens [31].

relevant information. AR applications are widely used for gaming. One of the well-known game in AR application is Pokémon Go (Figure 2.11).

Fig. 2.10 A tourist using an AR application [32]

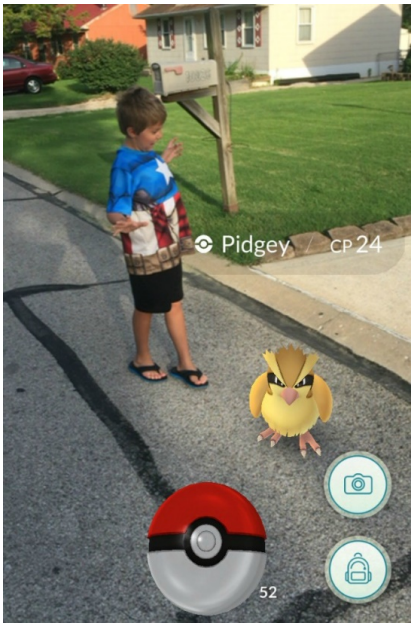


Fig. 2.11 A screenshot of Pokémon Go game [33]

2.6 AR in Maritime

Many maritime accidents are known to be caused by human error [34]. There is a range of advanced notification methods and equipment available nowadays. However, they have

not made a significant impact towards a reduction in maritime accidents [35]. The reason is that the provided information is too much for a human to interpret. It is important that the information is organized and presented in an interpretable fashion. Here AR technology can make a difference.

With the help of AR application in maritime, navigators can understand the information more easily as well as make decisions correctly and promptly. Figure 2.12 shows a conceptual sketch of bridge and implementation of AR technology.

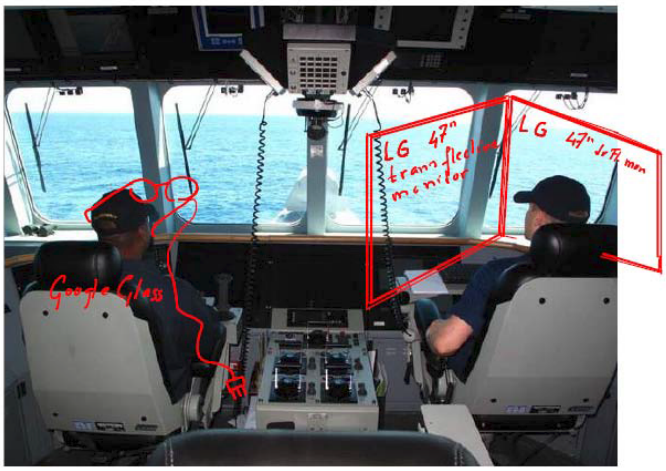


Fig. 2.12 A conceptual sketch of bridge and implementation of AR technology [36]

The literature review gives the theoretical and empirical support for this project. In this project, an inclusive AR-enriched application demo is developed for introducing the Ship Bridge.

Chapter 3

Methodology

This chapter discusses the methodology behind the development and design of an application for Microsoft® HoloLens augmented reality (AR) device. The Microsoft® HoloLens Application (App) Development consists of Tracking Systems, Marker-based Tracking, Camera and Marker Relationship, Camera Calibration, Marker Choice and Design, and Marker Training. The Application (App) Design is based on Design Concept, Design in Unity and Visual Studio, and Information Presentation Design. Each of the above is discussed in detail below.

3.1 Microsoft® HoloLens

In this project, Microsoft® HoloLens is the given device for development of the augmented reality (AR) application. It is a standalone device, which means that it consists of all computational accessories. It is the first holographic computer running Windows 10 operating system. The term 'HoloLens' is a combination of two words 'Hologram' and 'Lens'. A hologram is a virtual object that is entirely made of light and display on the goggles. The lens is a transmissive optical device that affects the focus of a light beam through refraction. This section provides HoloLens hardware and software specifications. In addition tools available for development of applications (app) are also discussed.

3.1.1 Hardware Specifications

Microsoft® HoloLens is a 3D holographic projection device with optical see-through head-mounted display (OHMD). Figure 3.1 shows the conceptual design of the device. It is a completely unattached; there are no wires, phones, or connections to a PC. Inside the HoloLens, it has cameras, computer, lenses, vent, sensors, and buttons. Figure 3.2 shows the

field of view (FoV) from a HoloLens. Figure 3.3 shows the components of the HoloLens from different views.

HoloLens is the world's first fully untethered holographic computer. Unlike the conventional computers, it has three processors such as: CPU, GPU and Microsoft custom designed Holographic Processing Unit (HPU).

HPU is a coprocessor to process and integrate data from the sensors, as well as handling tasks such as spatial mapping, gesture recognition, and voice and speech recognition [37].

Moreover, the interface of HoloLens is different from other devices. Instead of moving the cursor with the mouse, it uses voice, gaze, and gestures as input commands. The hardware specifications are given in Table 3.1.

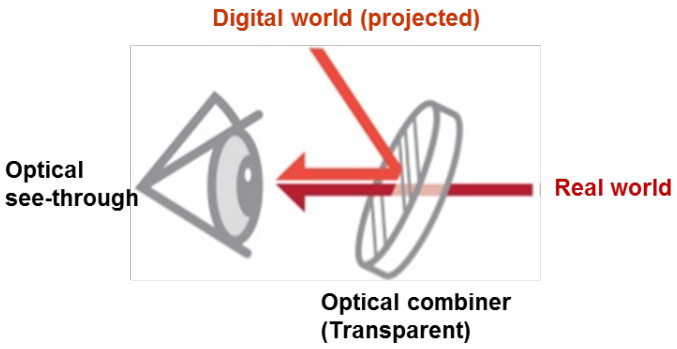


Fig. 3.1 Holographic projection OHMD design concept

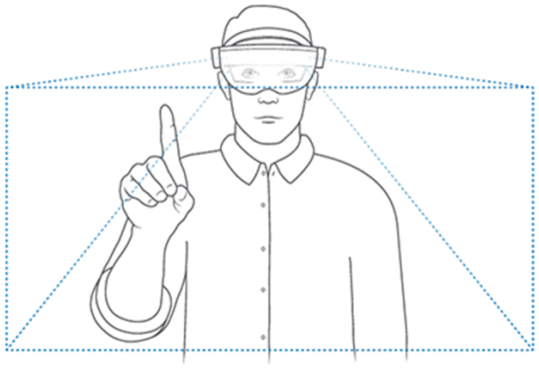
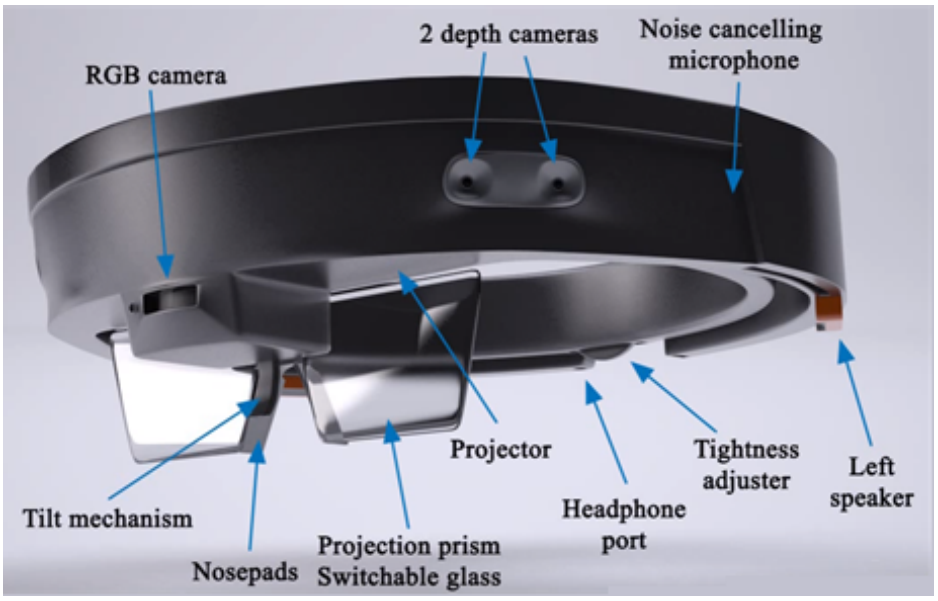


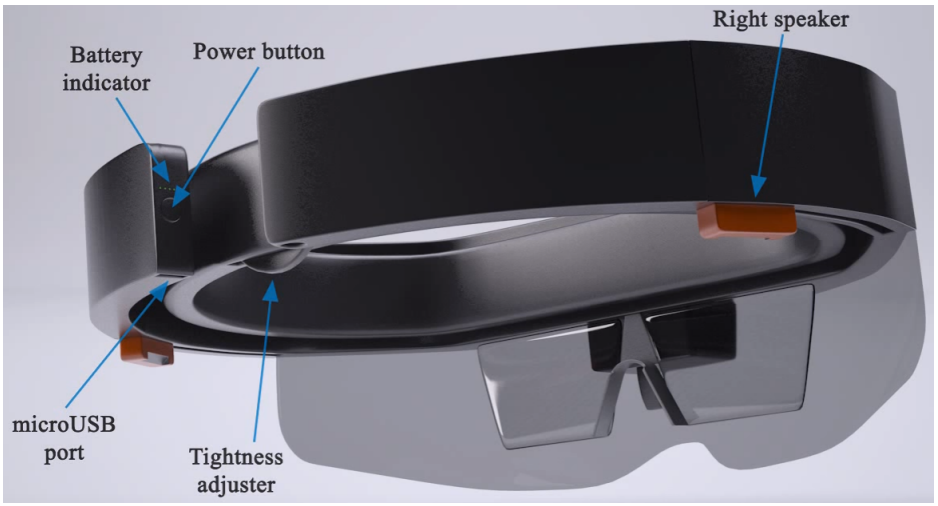
Fig. 3.2 Field of view from HoloLens [38]

3.1.2 Software Specifications

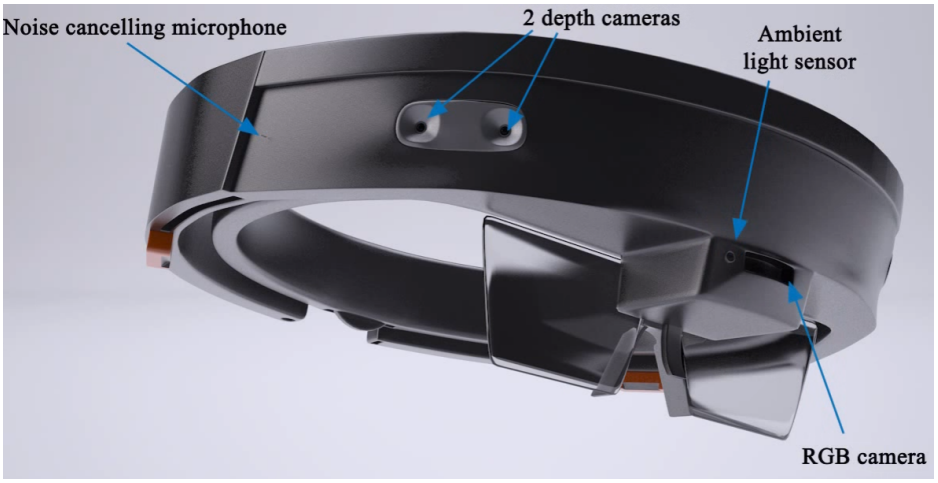
The software for Microsoft® HoloLens enables the user to fully interact with digital contents and holograms by applying five fundamental features of HoloLens: Gaze Control, Gesture



(a) Side view of HoloLens



(b) Back view of HoloLens



(c) Front view of HoloLens

Fig. 3.3 Shows the components of HoloLens from different views [39].

Microsoft HoloLens	
Operating system	Windows Mixed Reality
Processor	Holographic Processing Unit 1.0 (HPU), CPU, GPU
CPU	Intel 32-bit (1GHz)
Computing platform	Windows 10
Memory	2 GB RAM, 1 GB HPU RAM
Storage	64 GB (flash memory)
Display	2.3 megapixel widescreen stereoscopic head-mounted display
Camera	Photos: 2.4 MP, Video: 1.1MP
Audio	External speakers, 3.5mm audio jack
Input	Inertial measurement unit (Accelerometer, gyroscope, and magnetometer), 4 sensors, 1 120°×120° depth camera
Connectivity	Wi-Fi 802.11ac, Bluetooth 4.1 LE, Micro-USB 2.0
Battery	16,500mWh (2-3 hour active use battery life, 2 weeks standby, passive cooling)
Weight	579g (1.2lbs)

Table 3.1 HoloLens specifications [40]

Control, Voice Control, Spatial Sound and Spatial Mapping. The first three features are input for users which represent the primary ways to interact with the HoloLens. Spatial Sound recreates the sensation such that the sound is coming from a distinct location making virtual objects and sensations feel more real. Spatial Mapping accesses information about the surrounding environment. Figure 3.4 shows HoloLens scan and mesh the room by using Spatial Mapping feature.

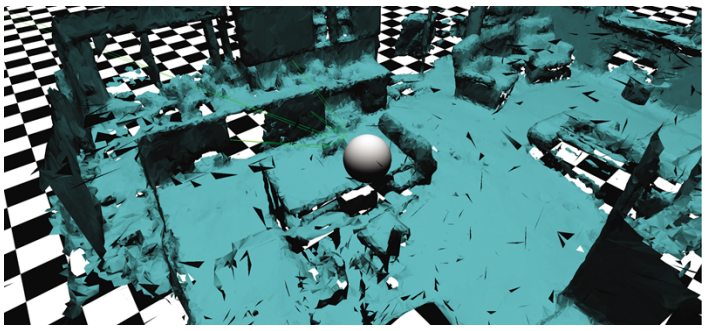


Fig. 3.4 HoloLens Spatial Mesh of a living Room [41]

The application of HoloLens could be training, education, design, gaming etc.. Figure 3.5 shows some of the applications for HoloLens. For example, in the education area, HoloLens can help students to increase their engagement and understanding of abstract concepts by blended physical objects and environments with 3D data.

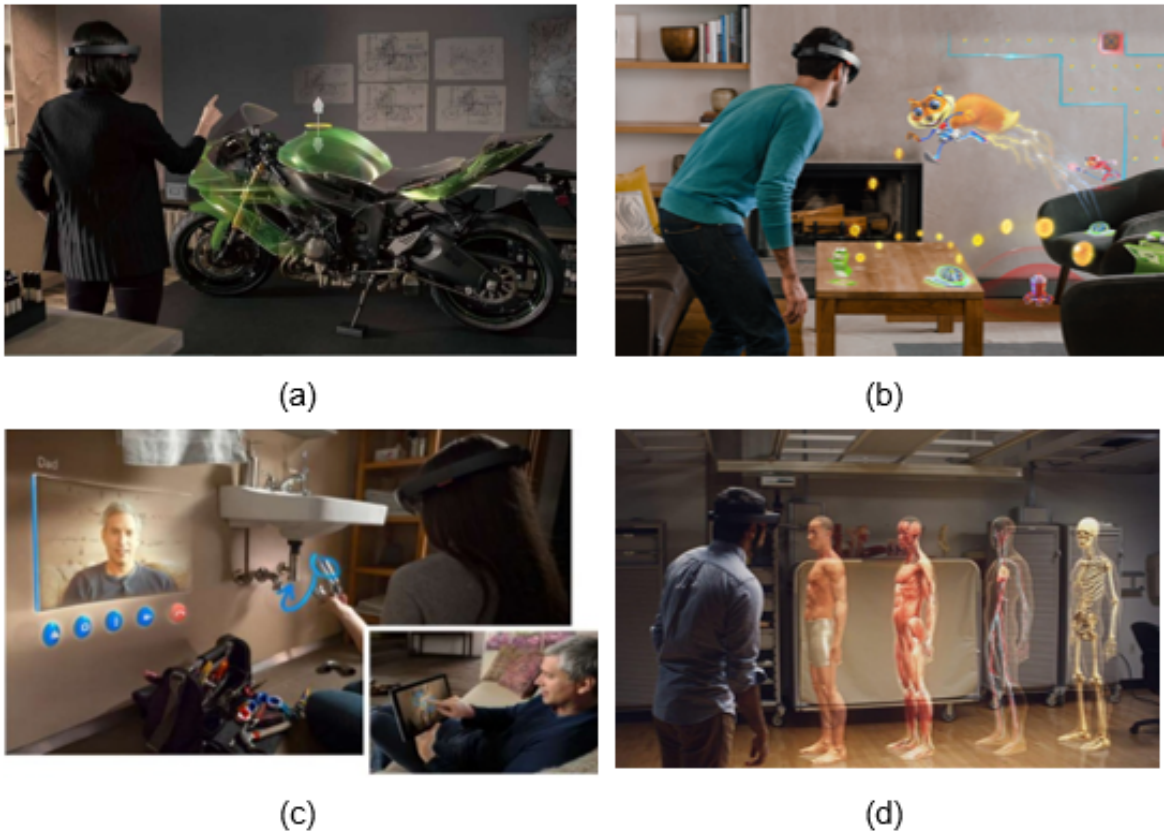


Fig. 3.5 (a) Designer is walking through and adjusting the 3D modelling rendered by HoloLens in real-time and real-space. (b) HoloLens for gaming. (c) HoloLens for remote instruction and sharing ideas (d) HoloLens for education [42].

3.1.3 HoloLens Application (App) Design and Development

HoloLens is using Universal Windows Platform (UWP) as an Operating System. Microsoft® has developed it especially for portable devices. Applications (apps) can be developed for HoloLens using Visual Studio with the help of Windows 10 SDK (version 1511 or later). It is to be noted that there is no separate SDK for HoloLens [43]. Building application for HoloLens needs following tools [43]:

- Visual Studio 2017 or Visual Studio 2015 Update 3 (Community, Professional, or Enterprise), needs for debugging and deploying.
- HoloLens Emulator (build 10.0.14393.0). allows the developer to test holographic apps on the PC without a physical HoloLens.
- Unity 5.5 or later. Unity is a cross-platform game engine which was recommended by Microsoft® for developing the holographic app.
- Vuforia® software development kit (SDK) for developing AR app.
- Microsoft® DirectX software development kit (SDK)
- Windows Device Portal for HoloLens. The Device Portal is a web server on your HoloLens that can configure and manage the device remotely over Wi-Fi or USB.

Application (Apps) development for Microsoft® HoloLens is discussed in detail in section 3.2.

3.2 Application (App) Development

3.2.1 Tracking Systems

In augmented reality (AR) applications, one of the fundamental components is finding corresponding points between the real world and virtual object projection. Finding the corresponding points requires tracking and recognition technologies. There are two methods to approach recognizing the points in the real world: marker-less tracking and marker-based tracking. Marker-less tracking can further be divided into model-based tracking and feature-based tracking.

The model-based tracking takes the idea of tracking objects as a whole and commonly use edges or lines of 3D model-based tracker for pose calculation [46] as shown in Figure

Fig. 3.6 An example of model-based tracking for complex objects [44]

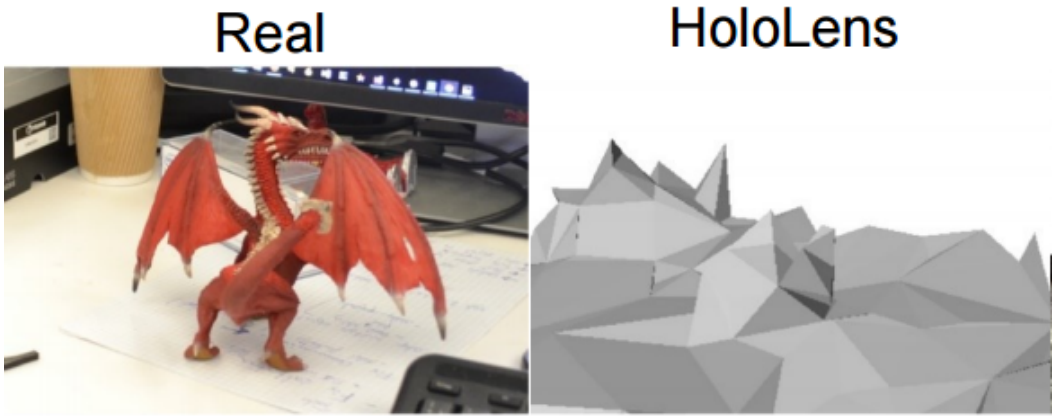


Fig. 3.7 Comparison of 3D data obtained from the Microsoft® HoloLens and real world [45].

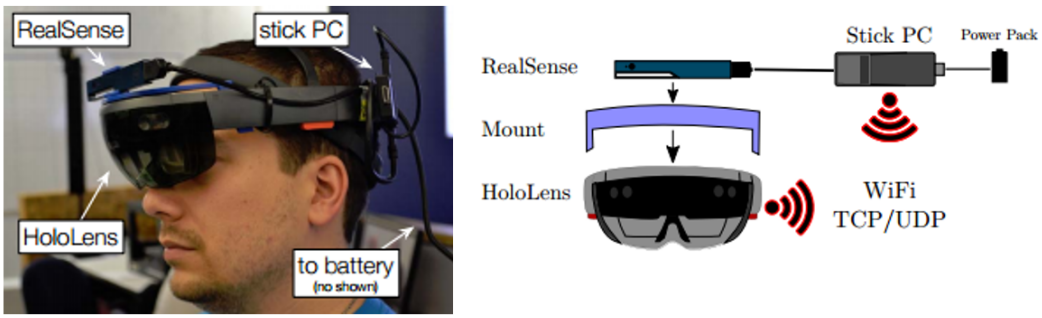


Fig. 3.8 Intel® RealSense integrated with HoloLens to get sufficient 3D data [45].

3.6. Object recognition is one of the applications of model-based tracking. Microsoft® HoloLens is not a suitable device to be used for model-based tracking, even though it provides 3D Spatial Mapping meshes to recognize and measure objects. The reason is that Microsoft® HoloLens provides a low-resolution scene reconstruction with low update rate (see Figure 3.7). Provided 3D data be insufficient for many applications with such as small scale object detection [45]. To detect relatively smaller objects, it needs raw depth data and RGB (red, green and blue colour) image from the front camera, but this data is not accessible to the developers with the application platform interface (API). To overcome this limitation, it is suggested to couple a high-resolution depth camera (for e.g. Intel® RealSense) with HoloLens and integrate with the onboard data (see Figure 3.8). This solution is too complicated. An alternative is to work with Object Recognition in Vuforia® library. However, there are limitations. Object Recognition to perform well in Vuforia® requires physical objects to meet certain conditions. In addition, the environment has to be supported.

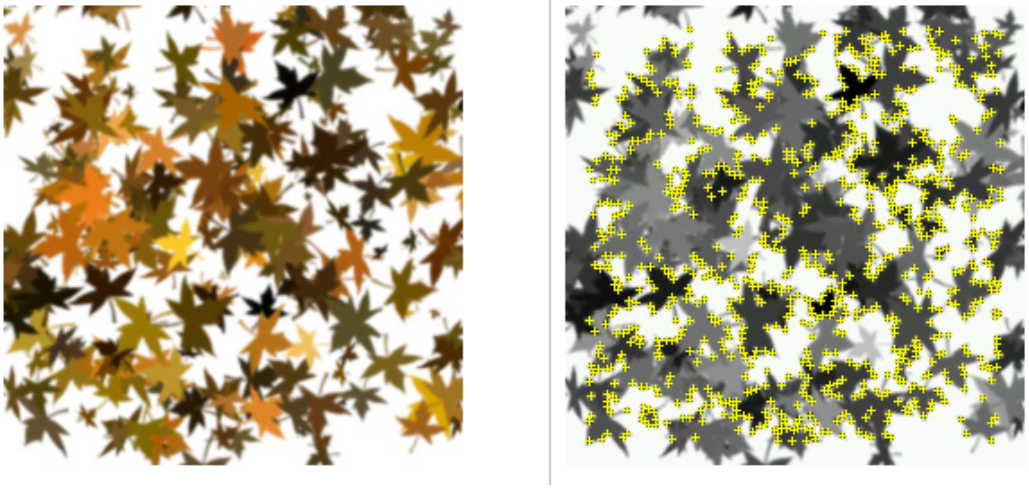


Fig. 3.9 Image with high number of features [47]

Feature-based tracking techniques recognize objects, not as a whole but track features such as distinguishable points or lines [48] as shown in Figure 3.9. Image target is one of the applications of feature-based tracking. In feature-based tracking, the feature of an image is compared with the features recorded earlier. This tracking mechanism works when features match with earlier stored data. However, if the environment has a number of similar shaped objects, for example, the panels in the Ship Bridge with a similar arrangement of buttons especially in the dark environment, feature-based tracking will not function. Because the panel has low local contrast and cannot provide enough richness in details to be detected and tracked so that the tracking performance will be poor.

Marker-based tracking uses markers as a target for tracking and detection (Figure 3.10). The marker is independent of the environment. The advantage of marker-based tracking is:

- Easy to track in a cluttered environment.
- Low cost because low-resolution camera may work.
- Easy Implementation with many available toolkits.

The disadvantages of using markers are that they might clutter the environment and they are not available for outdoor use. In this project, the tracking is indoor, and the Ship Bridge has enough space to put markers (Figure 3.11). The markers can be removed and replaced easily. In addition, they can be easily applied in a different Ship Bridge.

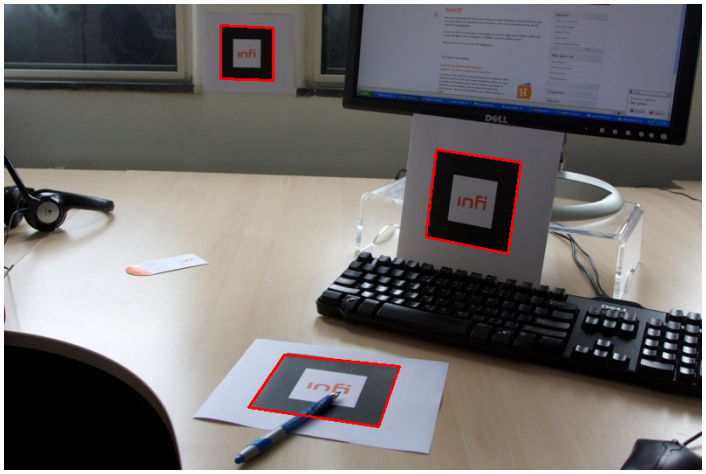


Fig. 3.10 Marker-based tracking [49]

Marker-less tracking that includes model-based tracking and feature-based tracking have certain disadvantages as discussed. Therefore, in this project, marker-based tracking is applied.

3.2.2 Marker-based Tracking in AR Application

Marker-based tracking is a common tracking method and widely used for visual tracking in AR [50–52]. Fiducial Markers (see Figure 3.12) have been commonly used in marker-based tracking systems. Fiducial markers are detected by a computer system by using image processing, pattern recognition and a variety of computer vision applications. Correct scale and pose of the camera can be defined by detecting fiducial markers. Scale and pose of the camera are then used to calculate the position and orientation of the marker.

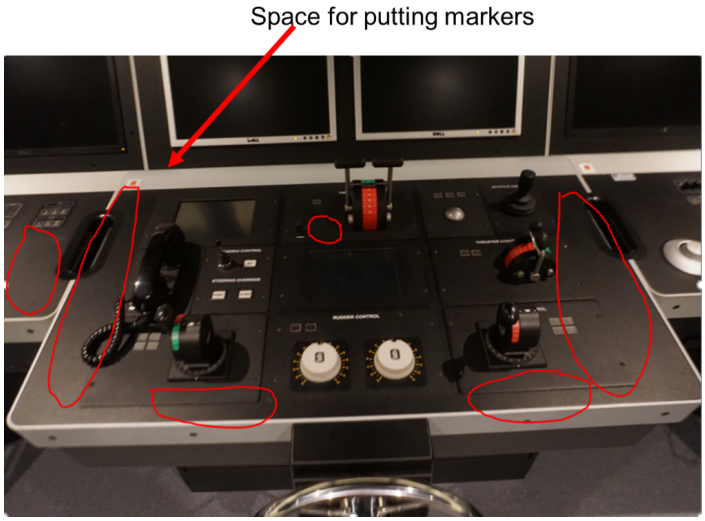


Fig. 3.11 Panel of the Ship Bridge

3.2.2.1 Selection of Marker-Based Toolkit

Fig. 3.12 Examples of fiducial markers [53]

There are many AR libraries for marker tracking and detection. Some of them are using circular markers (see Figure 3.13) while most of AR libraries work only with the square markers (see Figure 3.14). A circular maker with a single centre coordinate is not enough to derive the camera pose, whereas a square marker with four corner points can define the pose of the camera [53, 54]. It is preferred to use square based markers in most commonly used AR software development kits (SDKs).

There are two well-known SDKs for AR applications known as ARToolKit and Vuforia®. Both of them are relatively easy to implement [55]. When comparing the markers used in the two SDKs, AR marker (used in ARToolKit, see Figure 3.14) have certain advantages over VuMark (used in Vuforia®, see Figure 3.15). The advantages are:

- AR marker is a black and white square sign image which is easier to design than VuMark.

- Black and white based AR markers are easily and reliably detectable. It is because the higher contrast in the luminance makes the objects easily detectable.

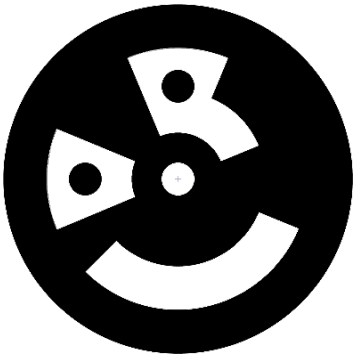


Fig. 3.13 An example of a circular marker [56]

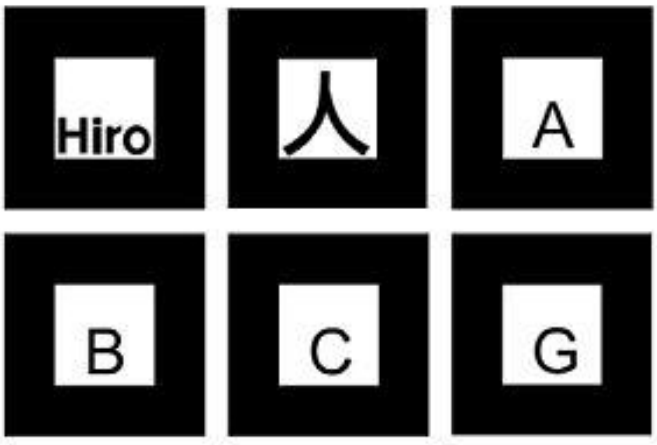


Fig. 3.14 Examples of square markers (also known as AR markers) [57]

Furthermore, when comparing the two SDKs, ARToolKit is an open source while Vuforia® is a commercial SDK. Vuforia® 6.1 (released in Nov 2016) supports Microsoft® HoloLens. However, according to some developers [59], it is still not the best choice. The reasons are listed below:

- It has a limited update rate.
- It is difficult to get everything that developer wants because of closed-source software.
- It costs to use.

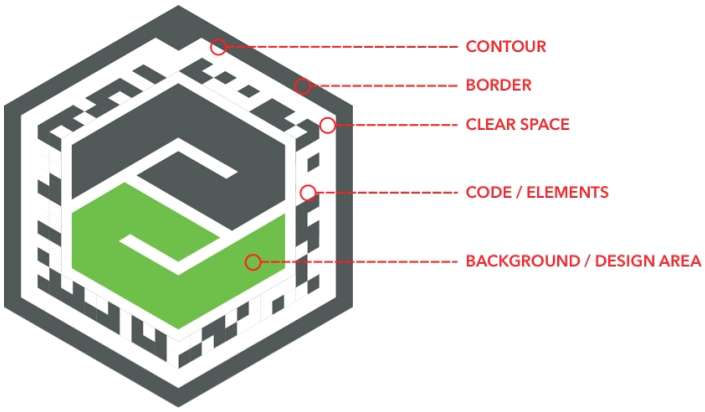


Fig. 3.15 Example of Vuforia® VuMark Marker [58]

ARToolKit is the first choice of the developers working with non-commercial AR applications [59]. One of the key obstacles of using ARToolKit is that it is not officially supported for Microsoft® HoloLens. Workers have looked to found a way around this obstacle, for example, [60] has shown that it is possible to use ARToolKit for the development of applications for Microsoft® HoloLens.

3.2.2.2 Implementation of ARToolKit Libraries in Microsoft® HoloLens

ARToolKit is written in C/C++ and has libraries which support marker based detection. C/C++ code is not supported in Unity game engine (used in this project) [60]. A way to do so is to convert ARToolKit libraries into Dynamic Link Library (*.dll*) file. This can be performed in Visual Studio software.

Figure 3.16 explains the procedure of linking ARToolKit in HoloLens. The procedure given in Figure 3.16 is explained as follows:

1. Download the required ARToolKit libraries.

2. Create a dynamic library project in Visual Studio, specify the name *ARToolKitUWP.dll*. Modify the **Additional Include Directories** property by copying the root path of the ARToolKit library package (Figure 3.17).

3. Add a C++ class in the dynamic library project,specify the name *ARToolKitUWP*. This C++ file is CLR Class Library that builds the native code (ARToolKit) using Common Language Runtime(CLR). This file connects the C# file and the ARToolKit libraries so that it can directly control the ARToolkit libraries [61]. The header file *ARToolKitUWP.h* defines the all the list of exportable functions.

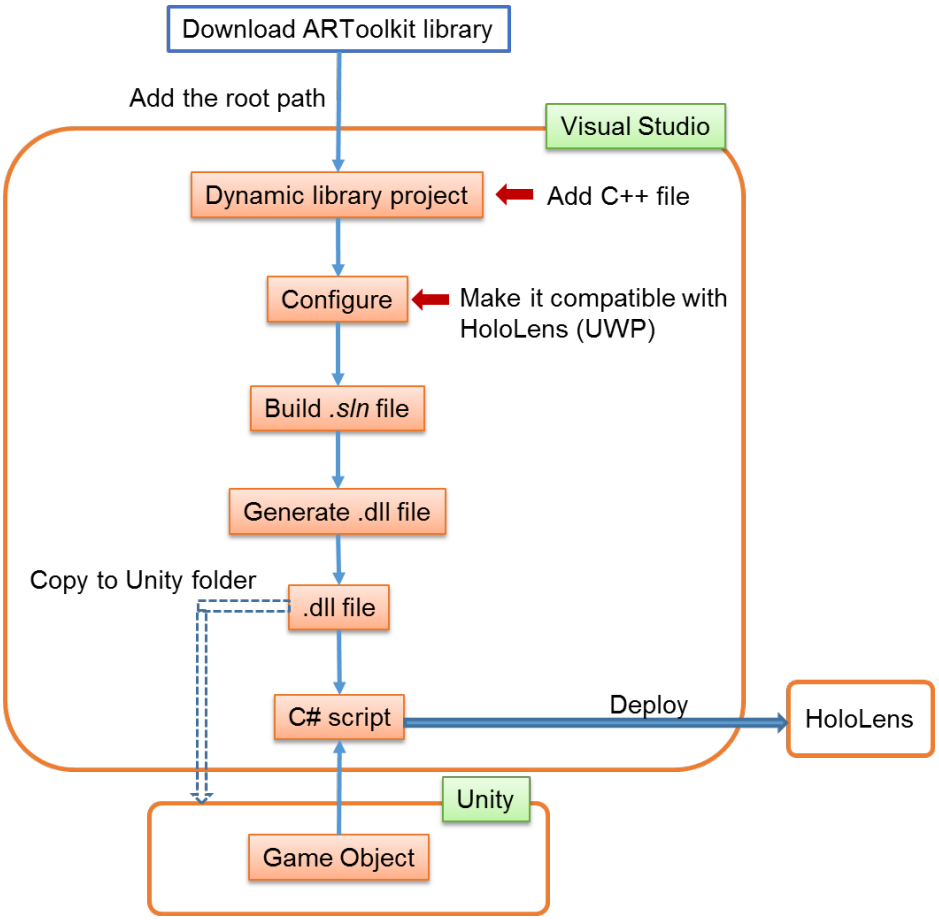


Fig. 3.16 Procedure of connecting of ARToolKit library with HoloLens

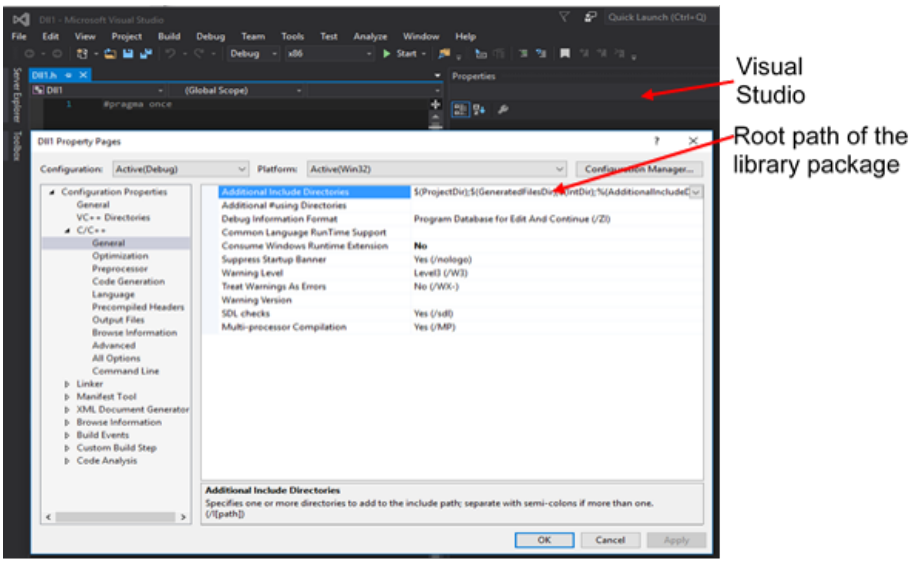


Fig. 3.17 Additional Include Directories property in Visual Studio

4. Some of the functions in the library cannot be called by outside, so an accessible interface exposed to these libraries. An extra C++ class should be included. It is a singleton class representing a controller called *ARController*, so that all the public functions of the singleton object could be called externally, and the native library manages one single object. In this project, the keywords to use that specifies exportable functions are extern extern "C" and __declspec(dllexport), and to call the function of detection and marker updates from *ARController* can be written in *ARToolKitUWP.h* as follow:

```
#define EXPORT_API __declspec(dllexport)
#include <ARController.h>
extern "C" {
EXPORT_API bool aruwpUpdate(AUint8* frame);
}
```

5. Afterwards, configure and build the project. It is important to change the options in Configuration Manager to make it is compatible with HoloLens (choose X86 for Active Solution Platform).

6. Finally, a native library (dynamic link library) *ARToolKitUWP.dll* is built by *ARToolKitUWP.sln* solution file in Visual Studio 2015 update 3 for x86. The library *ARToolKitUWP.dll* should depend only on UWP and WINRT system libraries, instead of general Win32 libraries (e.g. *kernel32.dll*). In order to successfully link the native library (ARToolKit) on the target platform (UWP), dependencies of a dynamic link library must be satisfied on the given device (HoloLens).

   The static library **.lib* is also create with the solution. It incorporates the following components of ARToolKit (v5.3.2) for detecting markers:

   - AR.lib
   - AR2.lib
   - ARICP.lib
   - ARMulti.lib

7. To use this native library for Unity and HoloLens, it needs to put the *ARToolKitUWP.dll* file in the Unity project folder: `Assets/Plugins/WSA/x86/`.

8. Create a C# script to import the *.dll* file which created above. Unity\C# builds managed libraries, but ARToolKit is unmanaged libraries (some are not native type of C#). Therefore additional work has to be done in order to correctly and safely interface managed and unmanaged libraries: Marshaling and PInvoke [60, 62]

9. C# file can be used in Unity. The native function entries are defined in *ARUWPNative.cs* file.

10. An example of calling a function from the library is given below:

```
using System.Runtime.InteropServices;     //system reference
public static class ARUWP {
    [DllImport("ARToolKitUWP.dll",
    CallingConvention = CallingConvention.Cdecl)]
    // provide the name of the library
public static extern void aruwpRegisterLogCallback
(ARUWPUtils.LogCallback callback);   // entry of function
}
```

The rest of the code in the file *ARUWPNative.cs* defines various constants used for tracking configuration in Unity.

There are five C# scripts in this wrapper, the rest of them are:

- *ARUWPMarker.cs*: defines a marker to be tracked by *ARToolKitUWP*, and relates tracking results with particular GameObjects in the scene.
- ARUWPMarkerEditor.cs: customizes the Unity editor itself , for the properties showing on inspector of Unity.
- *ARUWPController.cs*: configures the behavior of *ARToolKitUWP*.
- *ARUWPUtils.cs*: provides various utility functions serving the wrapper.

3.2.2.3 ARToolKit Algorithm

The algorithm of the ARToolKit tracking system is explaining as below:

- First, the front camera on the HoloLens captures video of the real world and sends it to the HoloLens.

- HoloLens converts the marker's frame to a binary image.
- The app on the HoloLens searches through each video frame for any square shapes (marker border) so that the black marker frame is identified.
- When the marker pose is found, the app will calculate the position of the camera relative to the marker's frame. Multiply by translation and rotation matrix can change the coordinate system from real world coordinate system to camera coordinate system.
- Once calculated the position of the camera, the marker needs to be identified via the symbols inside of the marker.
- When the symbol is matched with the templates in memory, the marker is identified, and the virtual object is aligned with the marker by using transform matrix.
- After the virtual object is rendered in the video frame, the final output is shown on the HoloLens.
- Through the display, the virtual object always appears overlaid on the tracking markers in the real world. The Figure 3.18 shows these steps.

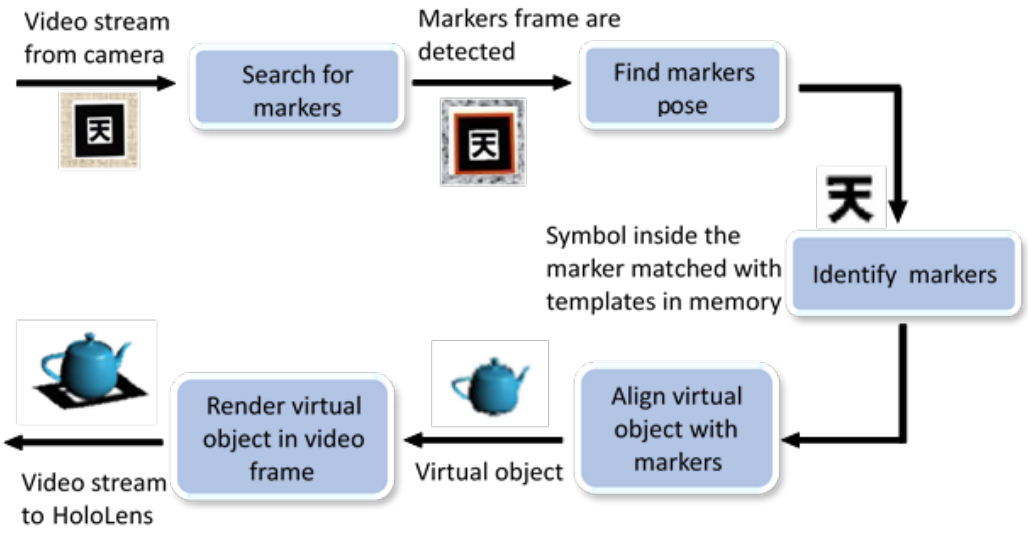


Fig. 3.18 Traditional square black and white AR marker tracking and detecting procedure.

3.2.3 Camera and Marker Relationship

In AR, the camera tracks the position of the marker in the scene. The display of HoloLens renders a virtual object aligned on the marker from the user view (Figure 3.19).

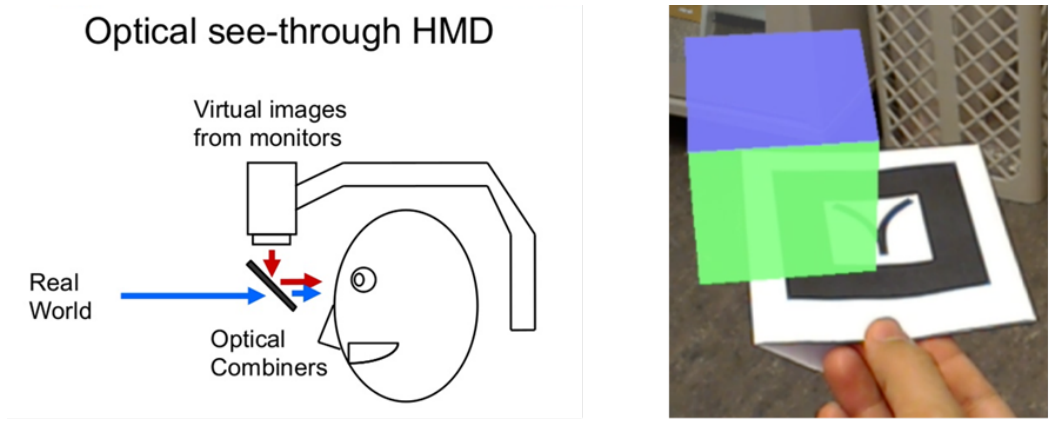


Fig. 3.19 (a) Conceptual of wearable AR devices and (b) User's view from the wearable AR devices.

3.2.3.1 Coordinate Systems

Figure 3.20 shows an overview of a coordinate system in HoloLens. In the tracking system, the transformation goes from world coordinates to camera coordinates. In transformation, all physical points are defined relatively to the camera. These points are used to render the pixels on the screen.

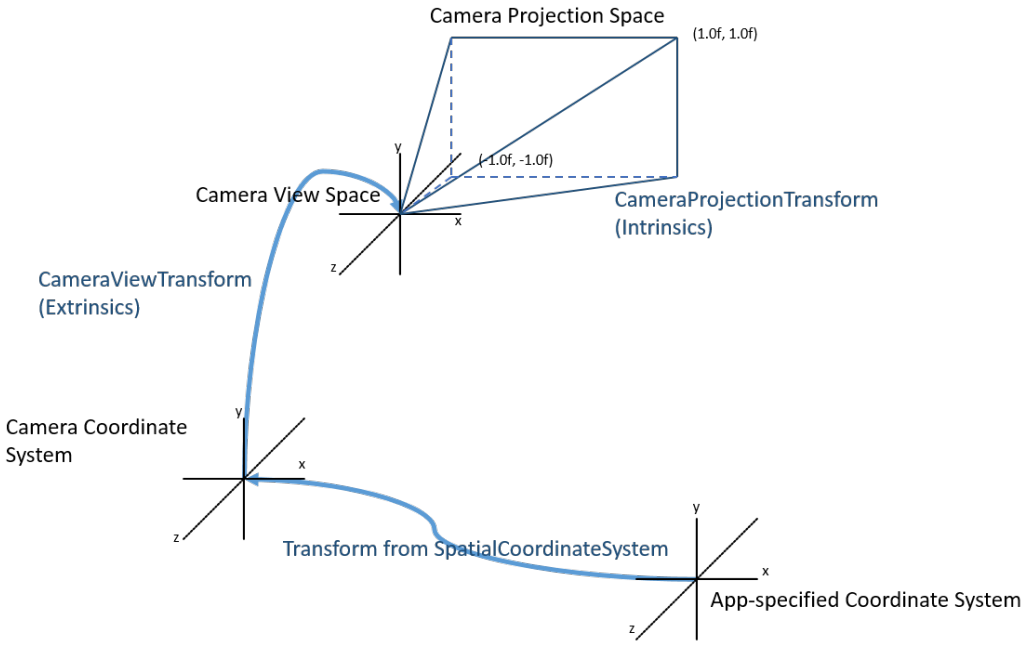


Fig. 3.20 Overview of coordinate system in HoloLens; work flow of locating a camera pixel from a single application [63].

In camera space, all points defined relatively to the camera. However, only the x and y coordinates are not enough to determine where an object should be put on the screen. The distance between the object to the camera (z direction)also needs to be counted. Therefore, homogeneous coordinates system (also called projective coordinates system) is introduced to express the distance between the points to the camera [64].

In homogeneous Space, all points are defined in a small cube. Everything inside the cube is onscreen. Multiplying everything by the Projection Matrix, the points will transform from camera coordinates to homogeneous Space.

In Figure 3.21 (a), blue cubes represent the real world's objects in the camera space, and the red shape represents the frustum of the camera. Every blue cubes inside the red frustum shape can be seen from the camera. This shows the view before multiplying with everything by the Projection Matrix. After multiplying everything by the Projection Matrix, the red frustum shape change to a cube with a size of -1 to 1 as seen in Figure 3.21 (b), and all blue objects are deformed to a frustum shape. Thus, the objects are getting smaller when they are further from the camera. This can be seen from 3.22 projection view of the red cube. There is another mathematical transformation that is to be applied in the shader to make the marker fit to the actual window size. 3.23 shows how the marker can be seen on the screen. Figure 3.24 shows the flow of marker transformation between different coordinate systems.

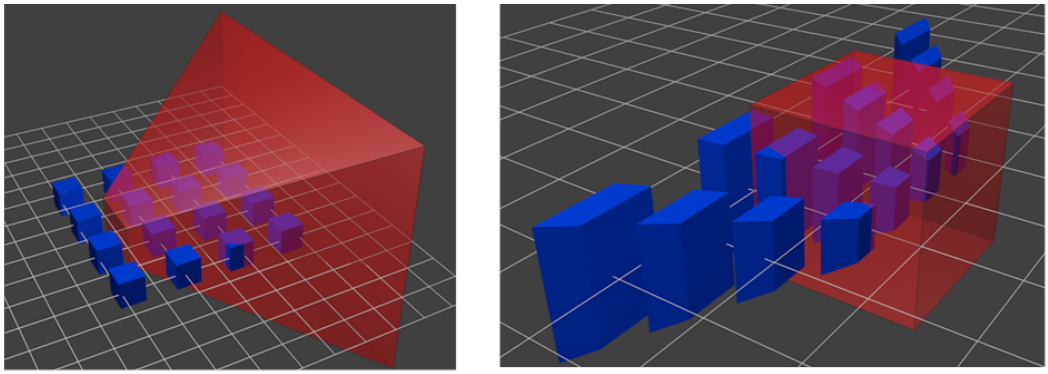


Fig. 3.21 The blue cubes represent objects in Camera Space, and the red shape represents the camera frustum (a) shows before multiplying with projection matrix (b) shows after multiplying with the projection matrix [64].

The main idea of AR is to present virtual objects in a real environment as if they are part of it [54]. Depending on the particular application, there are different tracking and pose estimation requirements. For instance, 3D graphic objects often need to be superimposed in good alignment with the real world. This requires the tracker to provide very accurate pose

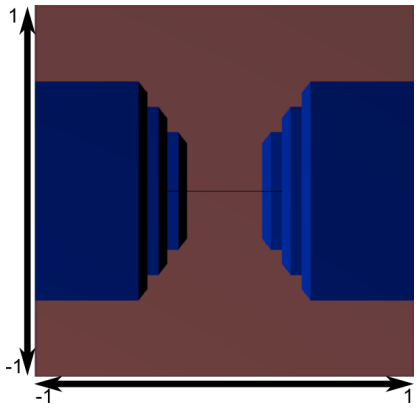


Fig. 3.22 Projection view of the red cube [64]

estimation. Pose estimation does not need to be very accurate to display the text information [65].

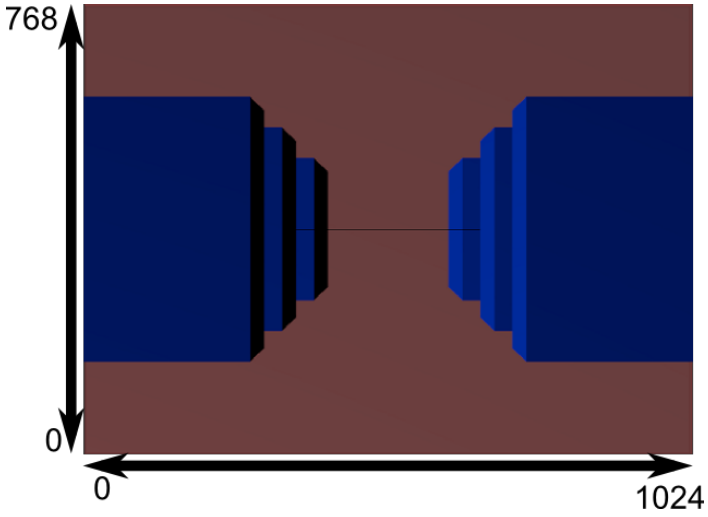


Fig. 3.23 Transformation is applied to fit this to the actual window size, and this is the image that is actually rendered [64].

There are two different types of coordinate systems: right-handed coordinate system and left-handed coordinate system (see Figure 3.25). OpenGL (Open Graphic Library, used in ARToolkit) uses the right-handed coordinate system. While Unity uses left-handed coordinates system. Appropriate transformation is needed when graphical information is exchanged between OpenGL and Unity.

The image RGBA data for tracking is provided by Unity WebcamTexture object. Therefore, when feeding the RGBA array obtained from Unity to ARToolKit, the image is flipped upside-down. In order to get right position and orientation of the virtual object showing up on the marker, the position given in the Unity has to be mirrored to the real world space.

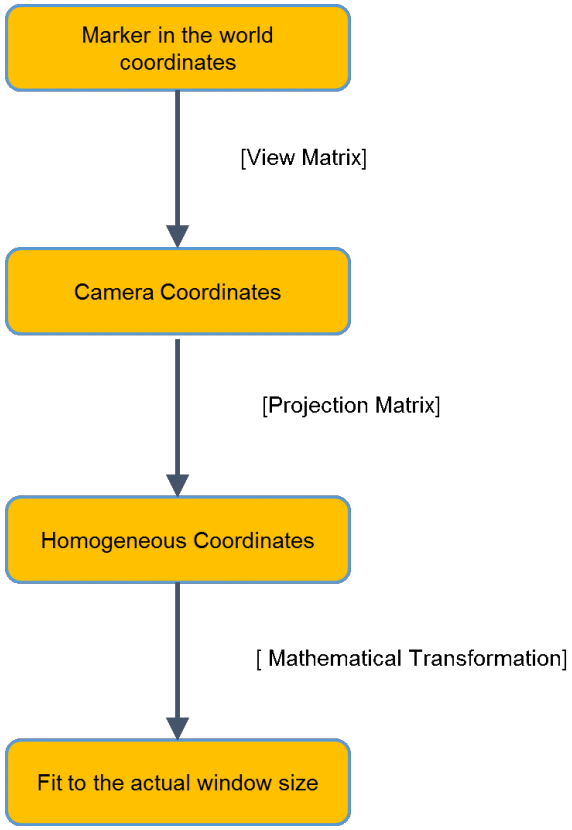


Fig. 3.24 The flow of marker renderer on the screen

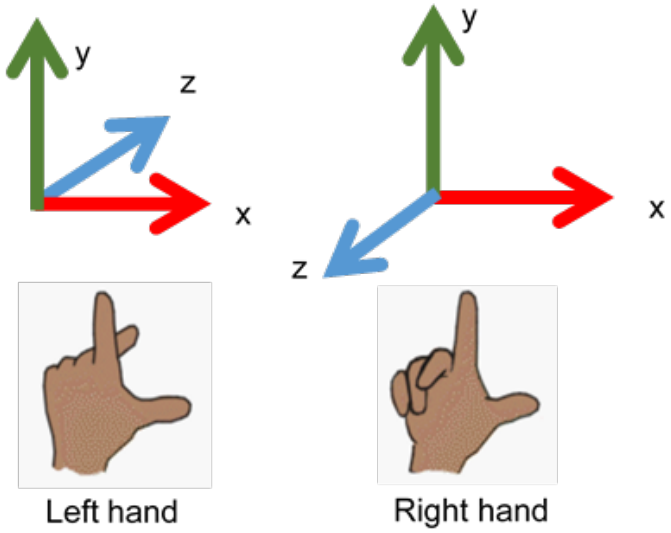


Fig. 3.25 Left\right-handed coordinate system

3.2.3.2 Calculations of Marker Transformation

In marker based tracking system, the marker is in the world coordinate system, while the camera has its own coordinates, see Figure 3.26. In ARToolKit, the point *X(x,y,z)* on the world coordinate system can be transformed by camera transformation matrix (the extrinsic camera matrix) **T** to get the projection of the point *p(u,v,w)* on ideal image coordinates. When HoloLens detects a marker, the extrinsic camera matrix **T** needs to be solved in the marker tracking system. The transformation matrix **T** includes translation vector $\vec{t}$ and 3×3 rotation matrix **R**. This can be expressed by Equation (3.1):

$$\vec{p} = TX = [R \mid \vec{t}]X \tag{3.1}$$

where $\vec{p}$ is the projection of the point, T is camera transformation matrix, X is the point on the world coordinate system, R is the 3×3 rotation matrix and $\vec{t}$ is the translation vector.

When writing in homogeneous coordinates, this expression can be written in matrix form as below (see Equation (3.2):

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.2}$$

where (u, v, w) is the image coordinate system, (X, Y, Z) is the world coordinate system.

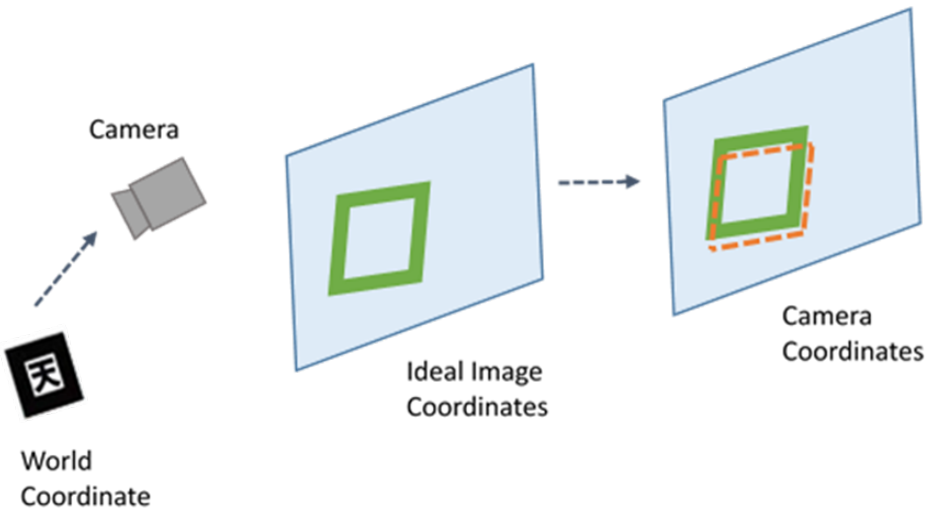


Fig. 3.26 Coordinate system in marker based tracking

In the tracking system, HoloLens is using the front mounted camera as the world-facing camera. It is a real-world RGB camera that can be simplified as a pinhole camera model [66]. Any pinhole camera may produce systematic geometrical errors and distortions due to lens imperfections. Therefore, an intrinsic camera calibration matrix **K** needs to be found out before the tracking starts. It is a mapping between ideal image coordinate system and camera coordinate system. The intrinsic matrix **K** is parameterized by Hartley and Zisserman [67] as below (Equation (3.3)):

$$K = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

where f_x is the focal length of the camera in the x direction, f_y is the focal length of the camera in y direction, (c_x, c_y) is the principal point at image centre, s is axis skew causes shear distortion in the projected image. In this situation, $s = 0$ since pixels are square and columns and rows are straight.

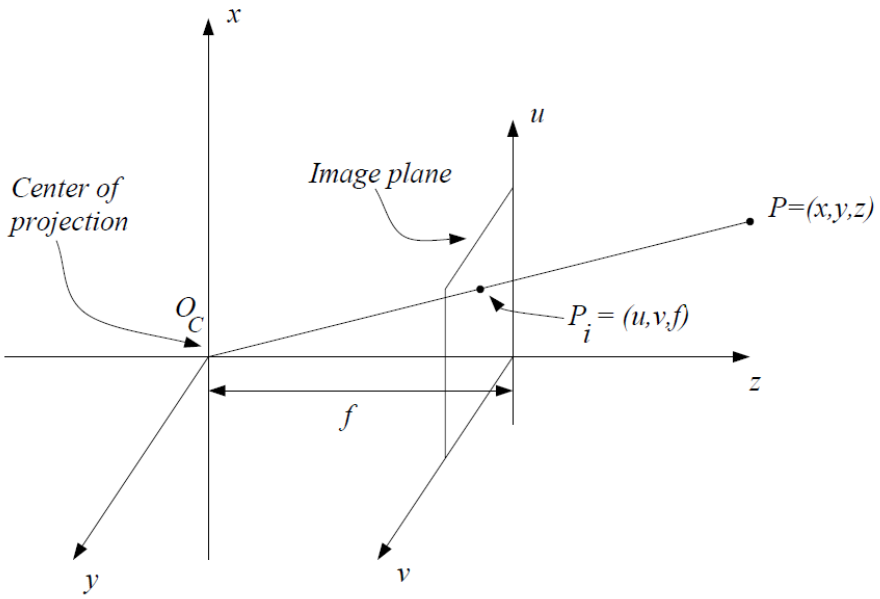


Fig. 3.27 The geometry of the simple pinhole camera model for perspective transformation [68].

Camera lenses also have some distortion including mostly radial distortion and slight tangential distortion (Figure 3.27). Therefore, distortion function is introduced and modelled afterwards. The distortion function converts the camera coordinates into camera's natural

units: pixels. In the end, the relation between the world coordinates and observed image coordinate system can be expressed as follow (see Equation (3.4) and (3.5)):

$$\vec{p} = D(K[R \mid t]X) \tag{3.4}$$

or written in matrix form:

$$p\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = D\left(\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}\right) \tag{3.5}$$

where D is the distortion function.

When it is radial distortion, distortion function D can be solved as Equation (3.6) and Equation (3.7). The results $(x_{corrected}, y_{corrected})$ is the corrected position on the output image for an old pixel point at (x, y) [69].

$$x_{corrected} = x\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{3.6}$$

$$y_{corrected} = y\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{3.7}$$

where k_1, k_2, k_3 are the coefficients, r is from rotation matrix.

When the image taking lens is not perfectly aligned to the imaging plane, tangential distortion occurs. The results of tangential distortion are given in Equation(3.8) and Equation (3.9) [69]:

$$x_{corrected} = x + [2p_1 xy + p_2\left(r^2 + 2x^2\right)] \tag{3.8}$$

$$y_{corrected} = y + [p_1\left(r^2 + 2y^2 + 2p_2 xy\right)] \tag{3.9}$$

where p_1, p_3 are the coefficients, r is from rotation matrix, x, y are from the word coordinate system.

In another word, solving the distortion function needs to find five distortion coefficients, which can be presented as a one-row matrix with 5 columns (Equation (3.10)) [69]:

$$D_{distortioncoefficients} = \begin{pmatrix} k_1 & k_2 & p_1 & p_2 & k_3 \end{pmatrix} \tag{3.10}$$

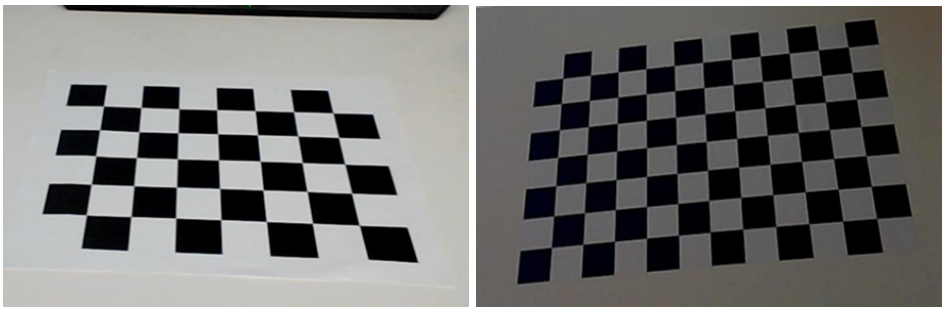
Where k_1, k_2, k_3 are radial distortion coefficients. p_1 and p_2 are tangential distortion coefficients.

In order to find all the five parameters, a well-defined pattern such as chess board image needs to be provided to find some specific points, for example, square corners in the chess board. As coordinates of the point in word coordinate system and coordinates in the image both are known, the distortion coefficients can be solved as a mathematical problem.

3.2.4 Camera Calibration

The calibration of the world-facing camera on HoloLens was done by the above model as discussed in section 3.2.3. The calibration file *calibrate.py* was a python file which was adapted from OpenCV Camera Calibration. In order to work for HoloLens, a C++ file for generating the binary camera calibration file for ARToolKit was used from HoloLensARToolKit package [60]. The calibration steps are as follows:

- Print out a standard chessboard pattern, for near range calibration it is 30cm (Figure 3.28 (a)) and for far range calibration, it is 45cm (Figure 3.28 (b)).



(a) (b)

Fig. 3.28 Chessboard images (a) for near range calibration with size 30cm; (b) for far range calibration with size 45cm.

- Capture the photos from the world-facing camera on HoloLens both in far range distance (1 to 1.5m) and near range distance (about 0.4m). At least 10 pictures in different angles of each distance are needed for the accuracy. These images are the input of configuration file in OpenCV.
- Find major patterns (corners of the squares on the chessboard) in the input. The function *cv2.findChessboardCorners (img,pattern_ size)* was used to find the pattern in chess board and the pattern size was passed in this function. The position of these corners of the squares would form the result.

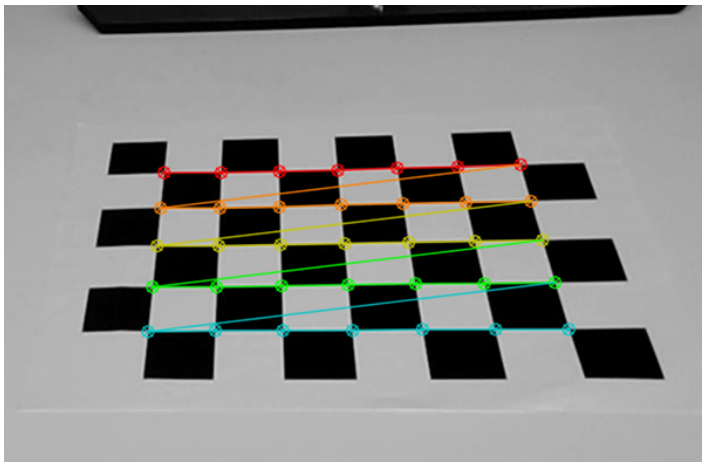


Fig. 3.29 Near range calibration pattern with $width \times height$ is 7×5 grid.

- Draw the pattern by using *cv2.drawChessboardCorners()*. Figure 3.29 and Figure 3.30 show images drawn on the patterns.

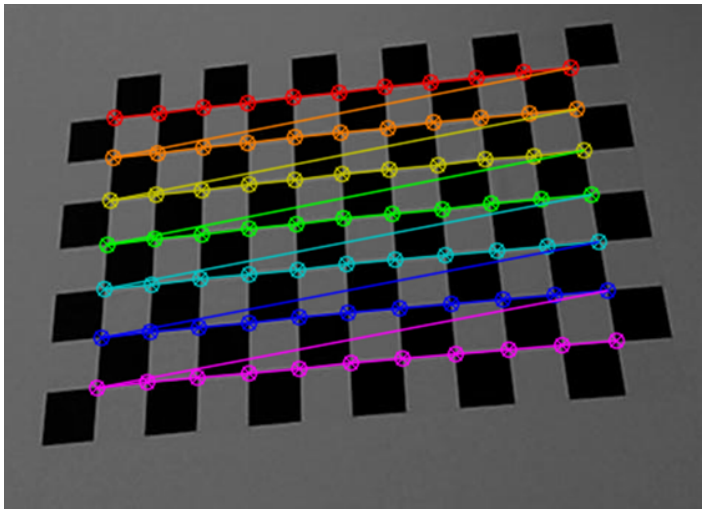


Fig. 3.30 Far range calibration pattern with $width \times height$ is 11×5 grid.

- Use OpenCV to determine the distortion matrix and camera matrix. These matrices are main component of the calibration file.
- Save the camera calibration files in OpenCV format (*.json* file and *.yaml* file).
- ARToolkit package is used to convert OpenCV format (*.json* file and *.yaml* file) to binary camera calibration format (*.dat*).

- The *.dat* file is to be attached to the Unity for marker tracking system.
- Use binary camera calibration file (*.dat* file) in ARToolKit and Unity. Unity read the camera calibration file in each time when the application starts.

It is important to use a high accuracy camera calibration file for AR applications. The reason is to get best tracking accuracy, avoid registration error, and reduce jitter when looking directly onto a flat marker. If the system does not have good camera calibration, the distortion error could be up to 10 pixels offset (roughly 11mm for holograms positioned 2 meters away). This could cause misalignment when a real-world object is viewed through HoloLens.

3.2.5 Marker Choice and Design

In ARToolKit tracking system, square markers with black border were used as AR markers. The AR markers have following advantages:

- Markers placed in the environment provide easily detectable visual cues for indoor tracking.
- It can be detected and identified in long distance at least 2 meters.
- The size of the marker is not too big to clutter the environment.

AR marker has its own rules of marker design in order to get good performance. For easy tracking, AR markers follow certain rules to design the shapes, border, background and marker image arrangement. There are two parts of a marker: components used for detection (border) and components used for identification (Figure 3.31).

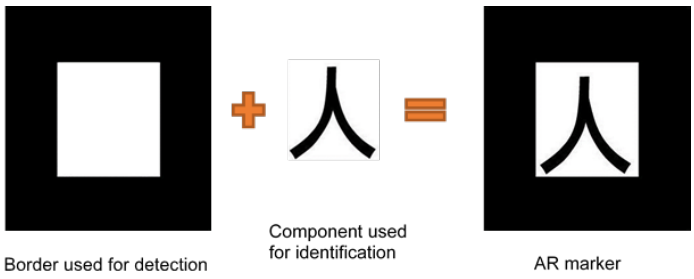


Fig. 3.31 Conception of AR marker components

The border is used to detect the marker, and it must be a contrasting colour to the background. Area of the black border needs to be large for higher accuracy of detection [70]. However, the white region has to be of a significant size for identification. It is recommended

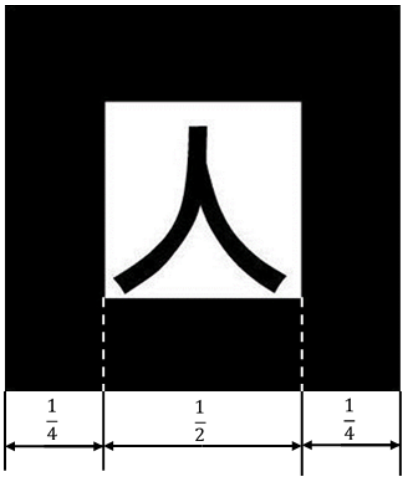


Fig. 3.32 Border requirement of AR marker

that to have the border thickness (black region) half the size of the inner marker (white region), see Figure Figure 3.32. For example, a marker of 80 mm will have the border thickness (black region) of 20 mm and the inner marker (white region) of 40mm.

Inner marker (white region) contains the identification signature. This could be a traditional AR template markers as shown in Figure 3.32. Or it can be a 2D-barcode (see Figure 3.33).

For the traditional AR template markers, the recognition component is an image as a template on the white background. The image can be black as well as coloured.

ARToolKit tracking system also supports 2D-barcode markers. For the 2D-barcode markers, the identifier is a unique matrix pattern which is predetermined.

The total number of possible barcode markers depends on the number of rows and columns in the matrix and the type of error detection and correction (EDC) algorithm. The number of possible barcode markers without EDC can be calculated as follow (Equation (3.11)):

$$n = 2^{n*n-3} = 64 \tag{3.11}$$

where n is the number of matrix in row and column.

For example, in a 3×3 2D-barcode markers, there are 64 rotationally unique pattern arrangements without EDC. The centre and the size of each cell need to be calculated to determine the ID of the marker. The value of each cell can be 0 or 1 (see Figure 3.33) so that the whole data of the marker can be presented as either a series of binary values or as a binary number [16].

When many markers are used at the same time, it is good to choose 2D-barcode markers. In this project, 2D-barcode markers were used for detecting the panel, and image template markers were used for detecting the stations. The image for template markers can be created in Adobe® Creative Cloud. The 2D-barcode markers can be generated by an online generator provided by ARToolKit [71].

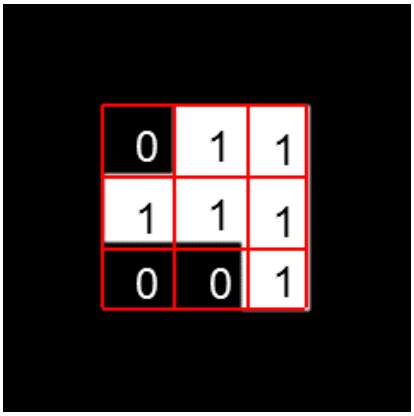


Fig. 3.33 Example of a 2D-barcode marker

3.2.6 Marker Training

ARToolKit has a marker "training" program called *(mk_ patt.exe)* to create the pattern file for recognizing the marker. It is used for creating template patterns for traditional AR template markers only.

This program only can be run in console mode. Once the program starts, the camera calibration file is requested (Figure 3.34). The program will open up a video window. The marker pattern file can be generated by following these steps:

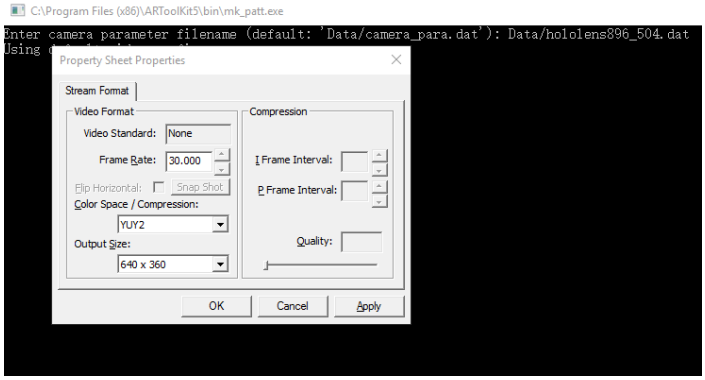


Fig. 3.34 The *mk_ patt.exe* program for marker training

- Put the marker on a flat surface in light conditions as similar as recognition application running environment.
- Adjust the camera to the marker. When a red and green square appears around the outer border, and a blue square shows up around the inner border, the marker is recognized (Figure 3.35).
- When the marker is recognized, hit the left mouse button to generate the pattern file. A name of the file will be automatically asked.
- After training all the needed markers, hit the right mouse button to quit the program.

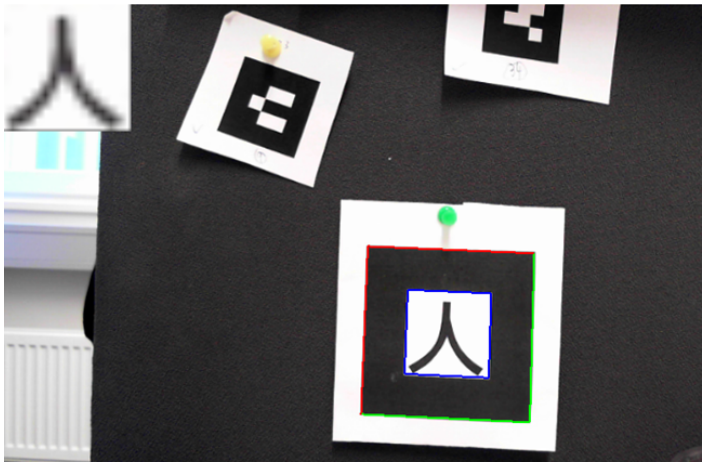


Fig. 3.35 The marker is recognized.

3.3 Appication (App) Design

3.3.1 Design Concept

The desired app was designed with HoloLens, using Unity 5.5.1f1 3D project and Visual Studio 2015. Marker tracking libraries from ARToolKit and HoloToolKit have been used. Figure 3.36 shows the basic interface and elements in Unity [72].

The design concept is based on the principle of user-friendliness and clarity. App has a "sense of purpose" therefore, it is easy to understand how to use it. The purpose of the App is given as: to familiarize the user (trainee) with the Ship Bridge and Maritime Operations.

The presented conceptual App has four scenes (see Figure 3.37). App function is discussed as follows: An introduction is given to the user on start of the app. After voice

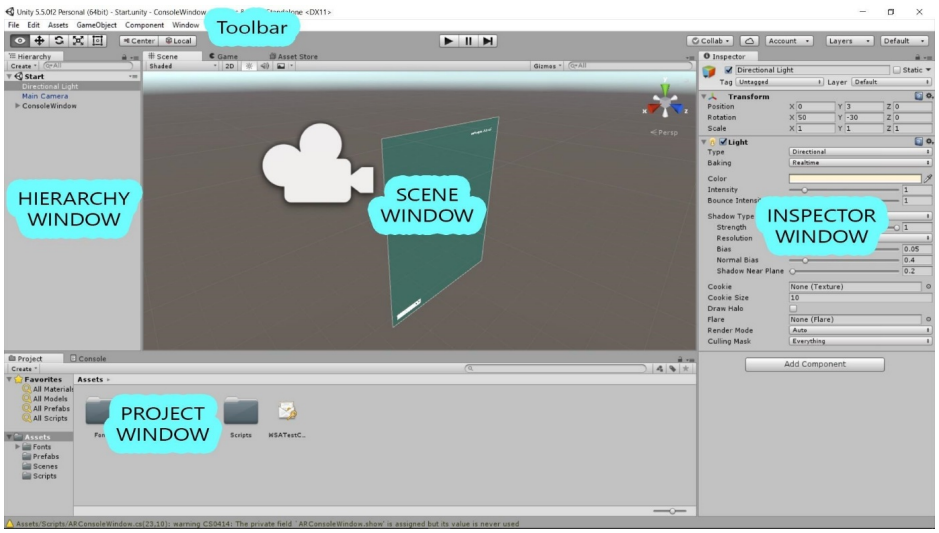


Fig. 3.36 The basic interface and elements in Unity [72]

command ”start”, the user will see the main menu. In the main menu, there are four options: Stations, Panel, Help and Quit. App functions are discussed below:

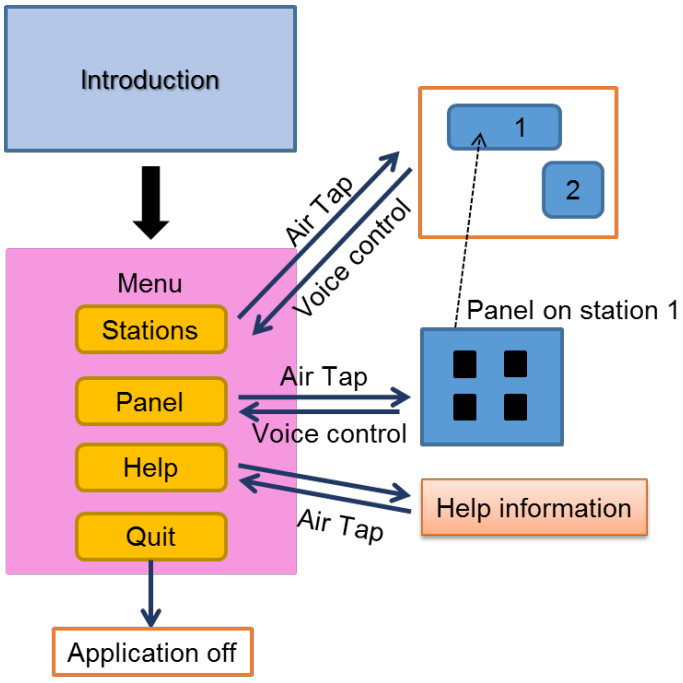


Fig. 3.37 App design concept

- Stations and Panel are the options for going to tracking scenes.

- When Air Tap to choose the Stations option, stations learning scene will show up. In this scene, AR traditional template markers are placed on each of the stations. When the markers are detected, the information will be given to the user.
- When Air Tap to choose the Panel option, the main steering station learning will show up. There are four 2D-barcode markers placed on the panel. The information will be shown when the markers are detected.
- By saying “Back”, it will bring the user back to the main menu.
- When Air Tap to choose the Help button, the help information and a back button will prompt up. Air Tapping the Back button will go back to the main menu.
- Air Tapping the Quit button will turn off the App.

3.3.2 Design in Unity and Visual Studio

Unity is recommended by Microsoft® as the AR program design tool for HoloLens. One Unity project consists of one or more than one scenes. The scene design is including hierarchy panel layout and adding components. Hierarchy panel layout means to put the game object in the scene. Adding components means to attach the components in the game object to make it functional.

3.3.2.1 Scene Design

In this project, scenes are designed in Unity with tracking and without tracking. Figure 3.38 shows the hierarchy panel layout for the ‘design scene with marker tracking’ and Figure 3.39 shows the hierarchy panel layout for the ‘design scene without marker tracking’.

There are two tracking scenes in this project: one is for stations, and another one is for the panel. Each of them has different types of marker for tracking. The reason for choosing image template AR marker for station tracking is because it works for long distance detection. Considering the real distance between the user and the markers, it is appropriate to have long distance markers for detection of stations. However, the distance of user while working on the panel is not as much. Therefore 2D-barcode markers are the better choice. 2D-barcode markers are easy to setup, fast to detect and have a lower probability of being mistaken for one another. In addition, they are recognized in constant time meaning that different 2D-barcode marker will take about the same amount of computer time to be recognized. Therefore, a large number of 2D-barcode marker can be used in a scene at little additional computational cost.

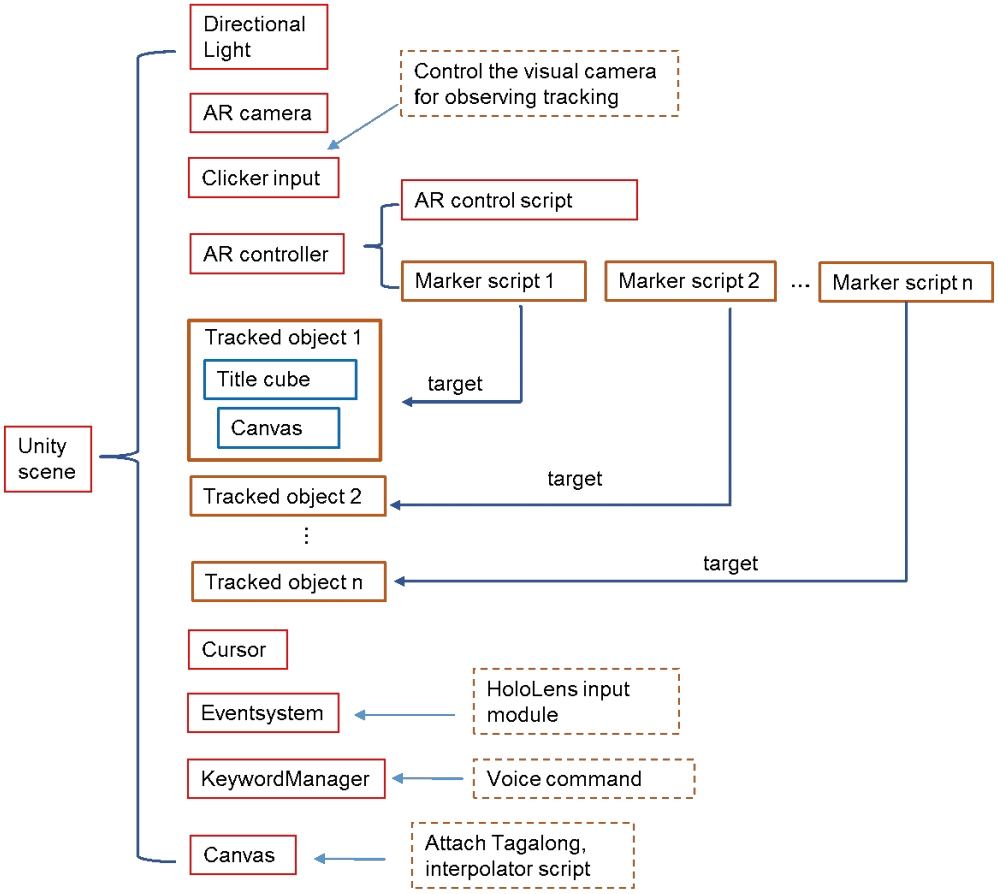


Fig. 3.38 Unity hierarchy panel layout for the ‘design scene with marker tracking’.

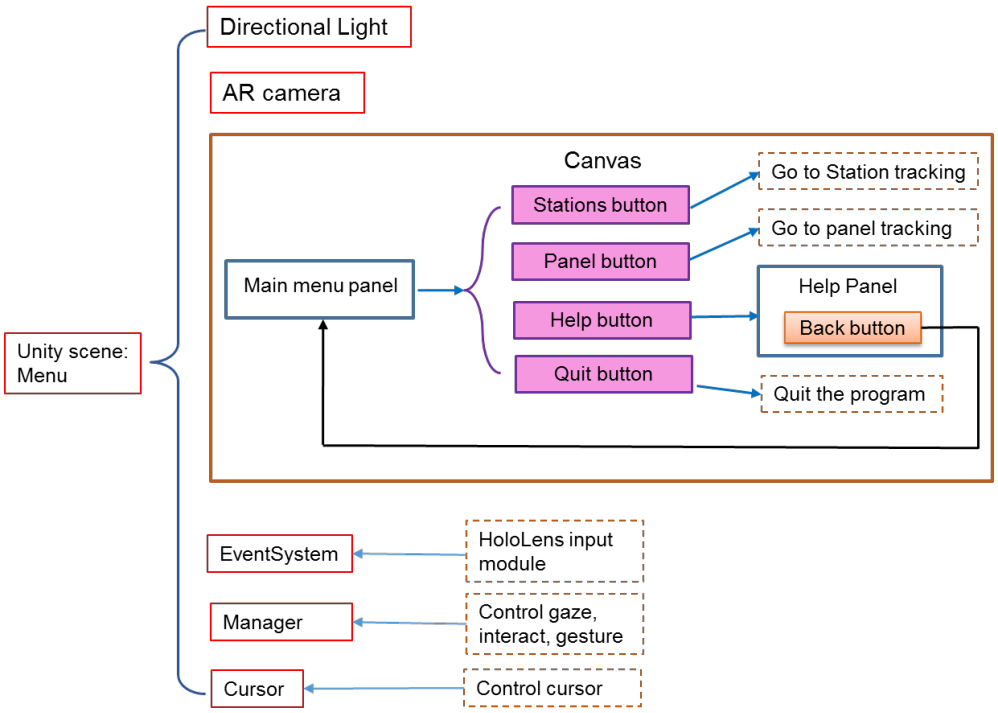


Fig. 3.39 Unity hierarchy panel layout for the 'design scene without marker tracking'.

One marker may represent one or more forms of information. The information must be the children of the Tracked Object. There are three ways to present the information: text information, playing video and 3D model animation. Figure 3.40 shows the hierarchy design for showing video and text information together in the tracked object. Figure 3.41 shows the hierarchy design for moving 3D models. It is important to set the relationship between each game objects, especially parental relation.

3.3.2.2 GameObject Design

GameObject is the foundation of the entire Unity project. GameObject with attached components make the project functional. There are some necessary GameObject(s) in every Unity scenes such as light and camera. The main camera is needed to be specified as an AR camera (virtual camera). It must be a child object of the AR scene root. The AR camera is the virtual camera and can be used to view the contents. The view shows the GameObject inside the frustum of the camera space. The tracked objects will be put outside the camera space as it will only be seen when the markers are detected. Figure 3.42 is the example of putting the GameObject in the virtual camera space.

There are some ready defined GameObject which developer can use directly, while many of them are needed to be modified to use. The scripts written in C# in Visual Studio are

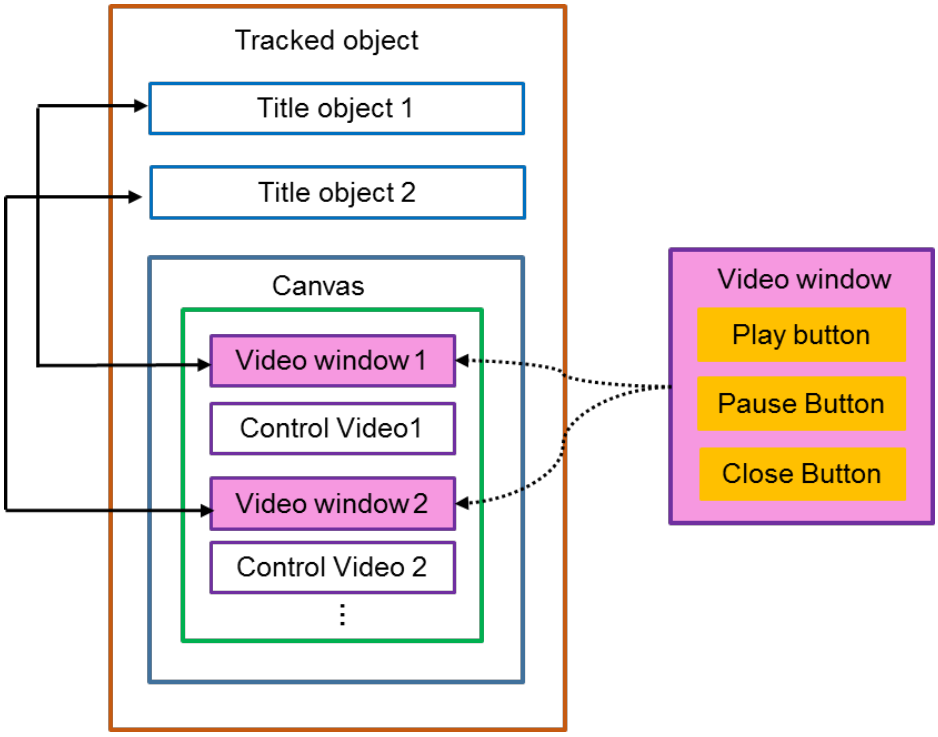


Fig. 3.40 One of the example of Tracked Object design

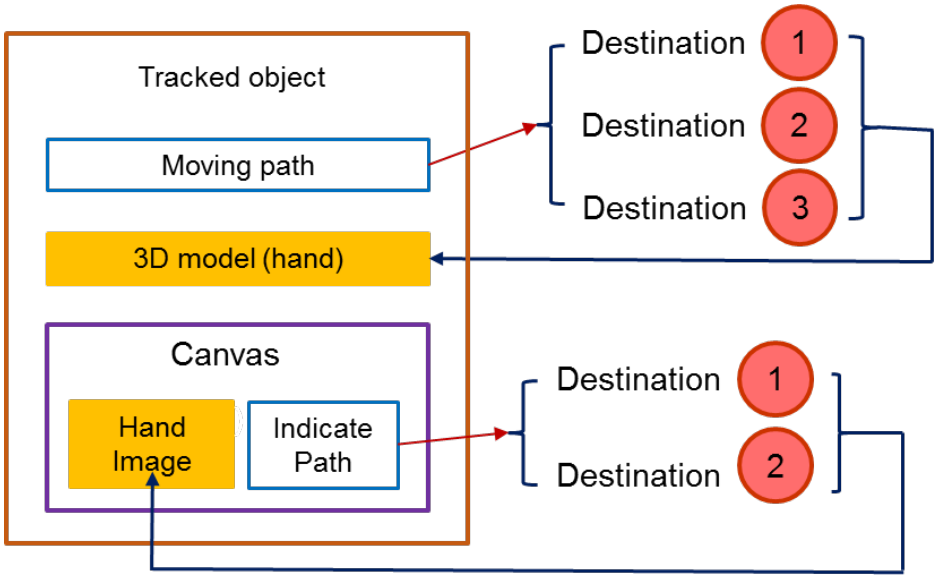


Fig. 3.41 Object Moving design

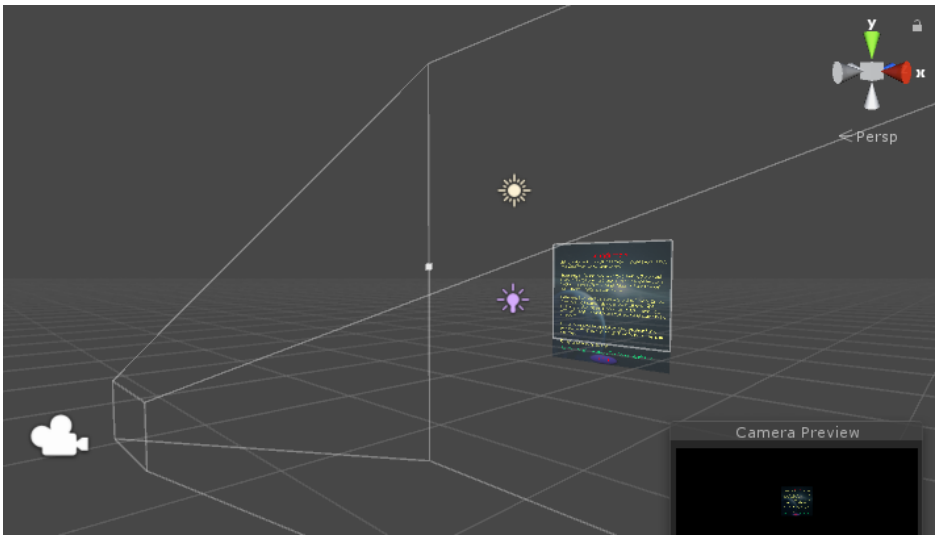


Fig. 3.42 Pose of AR camera and Scene in 3D view

considered as behavior components in Unity. They can be attached to the GameObject in the Unity scene. With other components in Unity, they can be applied to objects and are seen in the inspector. In this project, many functions are achieved by adding C# script.

3.3.2.3 Appliacation Functions Design

C# scripts were attached to obtain functions in Unity inspector. These functions are marker tracking, gaze control, voice command, gesture input, button behaviour, text shader, font material, movie player, and switching scenes.

- Tracking

  In order to obtain tracking function, two scripts were attached to the controller object. These were *ARUWPController.cs* and *ARUWPmarker.cs*. They are both from the HoloLensARToolKit package [60]. The main script *ARUWPController.cs* was the controller file. Each tracking scene was attached to one controller object. The controller object was at the scene root.

  – Options on Controller Script:

    Options on attached *ARUWPController.cs* C# script are shown in Figure 3.43.

    Camera Param is the name of the camera calibration file which was prepared before. This file must be stored in the Unity path `Assets/StreamingAssets/`.

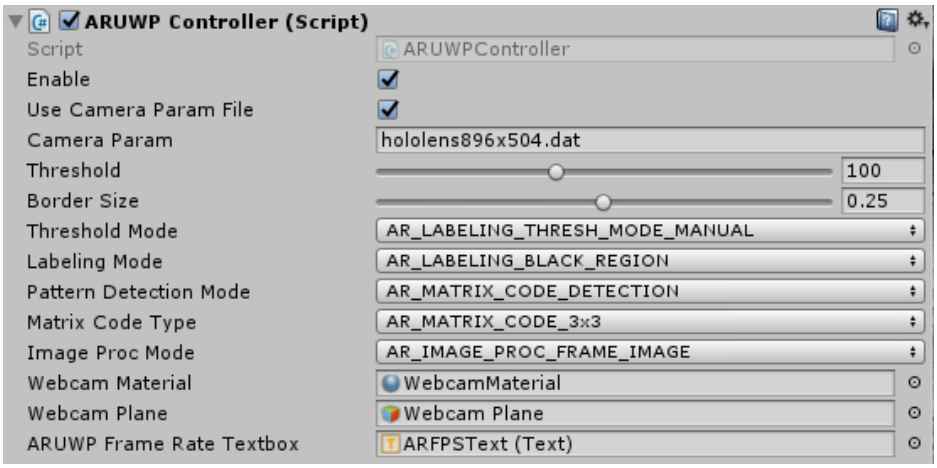


Fig. 3.43 *ARUWPController.cs* C# script controller options shown on Unity.

Threshold Mode represents a threshold method for the processing the marker. There are different threshold mode options. The threshold value is set according to the **Threshold Mode**.

Border Size is the percentage of the border of the marker. It depends on the marker which needs to be detected.

Labeling Mode configures the color of the border of the marker.

Pattern Detection Mode configures the marker detection algorithm. For example, AR_ TEMPLATE_ MATCHING_ COLOR or AR_ TEMPLATE_ MATCHING_ MONO is for image template markers. AR_ MATRIX_ CODE_ DETECTION is only for 2D-barcode markers. AR_ TEMPLATE_ MATCHING_ COLOR_ AND_ MATRIX or AR_ TEMPLATE_ MATCHING_ MONO_ AND_ MATRIX is for detecting both kinds of markers.

Image Proc Mode is the mode of image processing. For AR marker, frame image mode is used.

Webcam Material is a Unity Material object that for saving the frame data.

Webcam Plane is used for displaying the current frame on a Unity Plane object.

ARUWP Frame Rate Textbox is for showing frame rate information during tracking. It is for showing the system runtime status.

– Marker Script:

Tracking requires markers, and the number of marker scripts depends on how many markers need to be detected. Following are some important options:

Figure 3.44 shows the inspector window in Unity when marker script is attached.

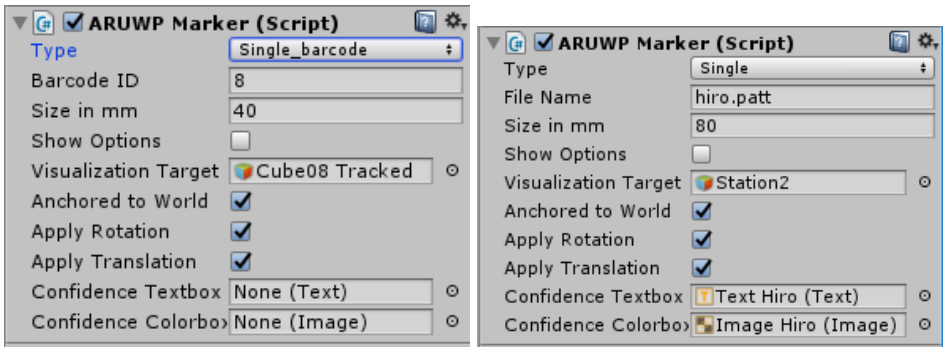


Fig. 3.44 Marker script options for different types of markers

Type is for selecting marker type. When the image template marker needs to be tracked, choose Single. The **File Name** option will appear. This is the file name of the marker pattern according to the marker. The matched pattern file must be in the root path of Unity path `Assets/StreamingAssets/`. When the 2D-barcode marker needs to be tracked, choose Single_ barcode. Then the **Barcode ID** will be required to differentiate it from the other markers.

Visualization Target connects with the detected marker and tracked object in the Unity scene. It controls the tracked object that receives the pose of the marker and creates augmented reality experience.

World anchor is a localization algorithm from HoloLens itself. It is added inside the marker script. When **Anchored to World** is true, the marker stays in the same positon, even the marker is occluded, or the HoloLens camera is blocked. The tracked object will not move in the world coordinate system.

- Mesh Filter

The HoloLens has depth cameras which continually scan the user's environment. These cameras create three-dimensional geometric representations of the objects in the environment. The HoloLens scans the environment to create a spatialize mesh representation of the physical world. It takes the raw mesh data and put into a format that the unity developers may use. A standard component in Unity called mesh filter component contains the mesh data.

A mesh filter component and a mesh render will be used together when rendering the object. A mesh filter component and a mesh collider component can be used for interacting with physical geometry and ray casting.

- Voice command

The voice command is one of the features of HoloLens. By using HoloToolkit, such features can be added in the designed application. The voice command is implemented in the C# script *KeywordsManagers.cs*. In this script, it specifies keywords and methods in the Unity (see Figure 3.45). It converts the keywords and the methods to the values so that the recognized keywords can be linked to the *UnityEvent* class. Once the recognized keyword exists, the corresponding method will be invoked.

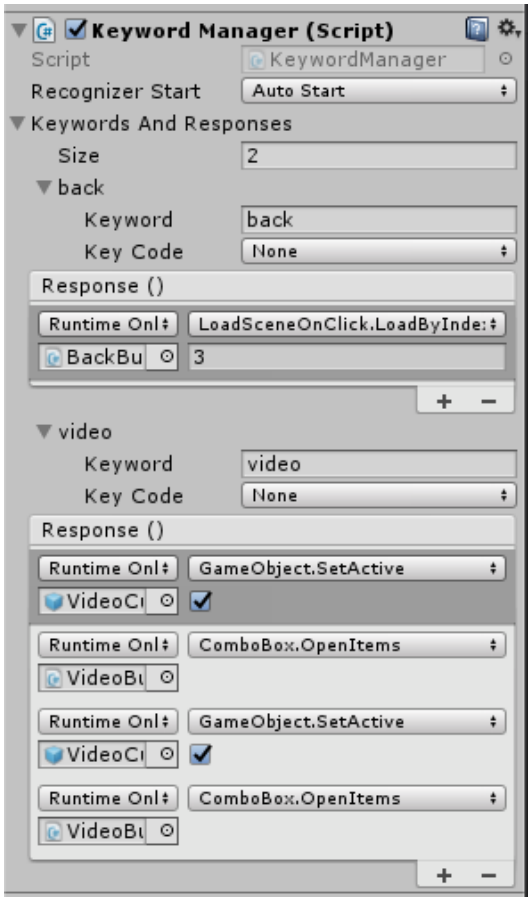


Fig. 3.45 Example of voice command settings

In this project, there are three voice commands: “Start”, “Back”, and “Video”.

1. “Start\Back” command: These two commands are for switching to another scene. The “Start” command is used in the introduction scene to go to the application menu. “Back” is used for going back to the main menu. When the command is recognized, the corresponding function in class *LoadSceneOnClick* will be invoked. The *LoadSceneOnClick* class is a C# script. It is attached to the “Start” button and the “Back” button.

2. "Video" command: "Video" command controls the video plane on and off. When the video is showing up, the command will let it disappear and vice versa. This function are approached by functions *SetActive()* and *OpenItems()*.

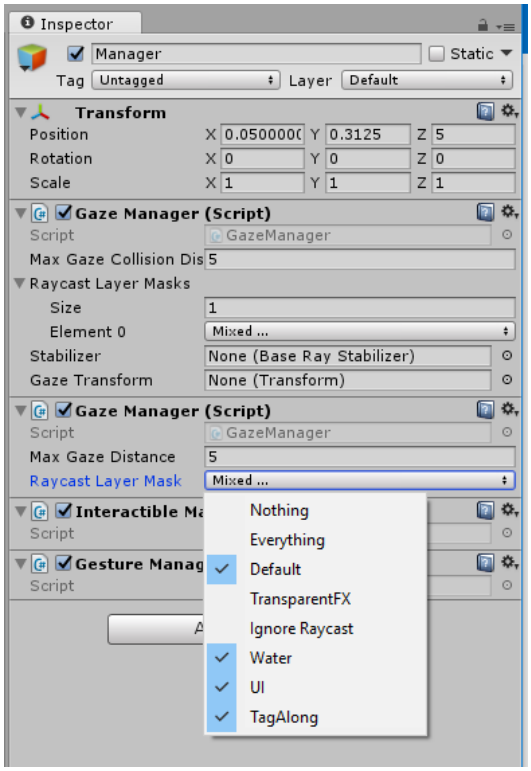


Fig. 3.46 The Inspector window for Manager object in Unity

- Cursor

The cursor is the object that follows the camera gaze. It has the same functions as a mouse cursor on a PC screen.

CursorManager class is from HoloToolkit. It has two parts 'on Holograms' and 'off Holograms'. It shows appropriate Cursor according to the desired Hologram position.

- Gaze Control and Gesture Manager

Gaze control and Gesture manager were written where functions were called from HoloToolkit. *Gaze Manager* class determines the location of the user's gaze, hit position and normal direction. *Gesture Manager* class contains event handlers for subscribed gestures. *Interactible Manager* class is called in the *Gesture Manager* for tracking and selecting the object.

Gaze Manager, *Interactible Manager*, and *Gesture Manager* Scripts can be attached to one GameObject called Manager (Figure 3.46): The Manager object is put under the scene root.

- Tagalong

The script *Tagalong* was written where functions were called from HoloToolkit. It allows the object in the application to follow the user. The script is attached to the GameObject in the Canvas which is the parent GameObject of the menu panel (Figure 3.47). Therefore, the menu follows and appears in front of the user.

Set up in the Unity is important to make the UI respond to the gaze and tap. A script *billboard* should be attached to the same GameObject with Tagalong. A new layer should be created in the Inspector Layer option and name it 'TagAlong'. Canvas object should be set (but not its children) to be on the 'TagAlong' layer. Then set the 'Raycast Layer Mask' on the *GazeManager* to ignore the 'TagAlong' layer as shown in Figure 3.46.

The Collider is a component in Unity that allows the GameObject to detect that it has intersected with another GameObject. A box collider is needed for gaze to respond. The size of the collider needs to fit around the UI elements.

In addition, *Render Mode* is set to *World Space*, and the *Event Camera* is set to the *Main Camera*. These settings are necessary to make gaze and cursor work.

- Button function

Several types of buttons were created for different purposes. The C# script *button behaviour* was written where functions were called from HoloToolkit. The function *OnSelect()* was called to trigger the button. It should be careful that the *OnSelect()* function does not work unless the gaze vector collides with the game object. The button object should be inside the collider. Based on this script, other scripts can be added under the same button to do a different job. For example, the 'play', 'pause' and 'close' buttons are created for making the button control the video. The *ComboBox* script can be added to control the GameObject activate and deactivate with the same button. The *QButton* class is attached to the button to quit the application. The example button Inspect from the project is shown in Figure 3.48.

- Video play

In order to give vivid information, showing video is always a good choice. In this project, the video is shown on a cube object. Since this project was initially created in Unity 5.5 however with release Unity 5.6 in April 2017, new steps were devised to Video Player component. These steps are given as follows:

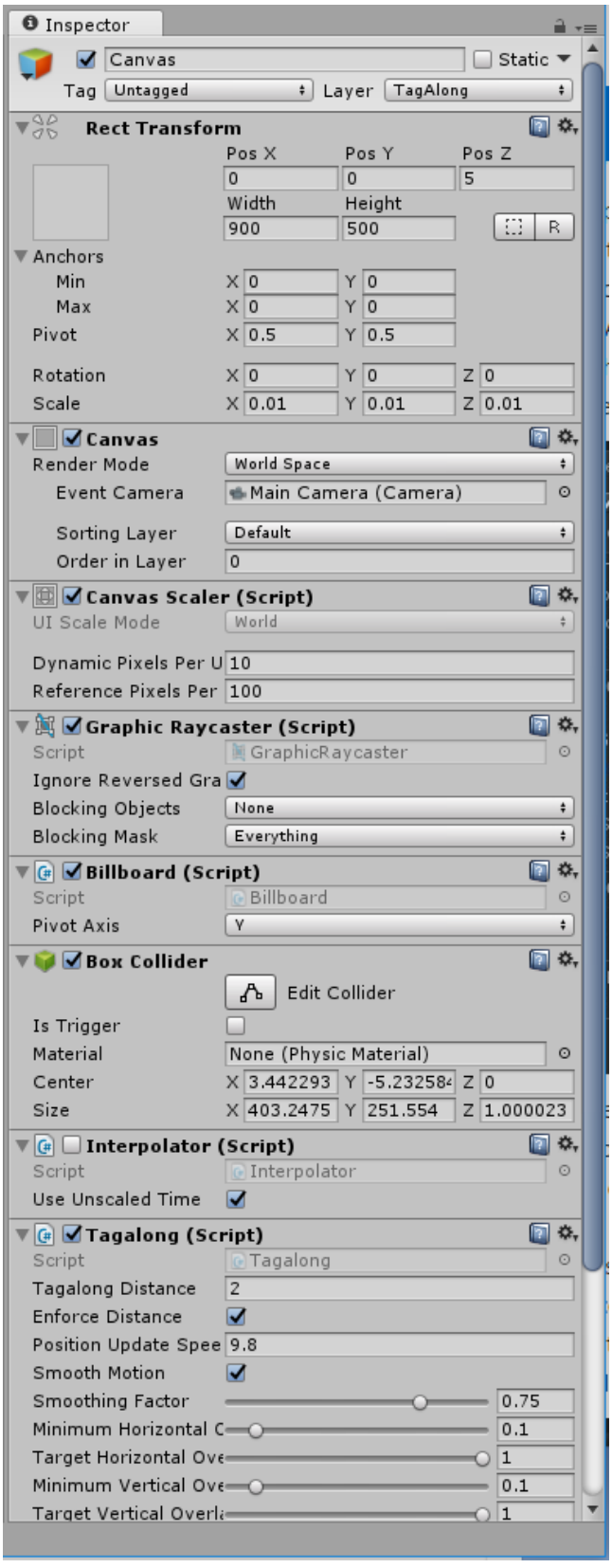


Fig. 3.47 Canvas appearance when *Tagalong* script is attached.

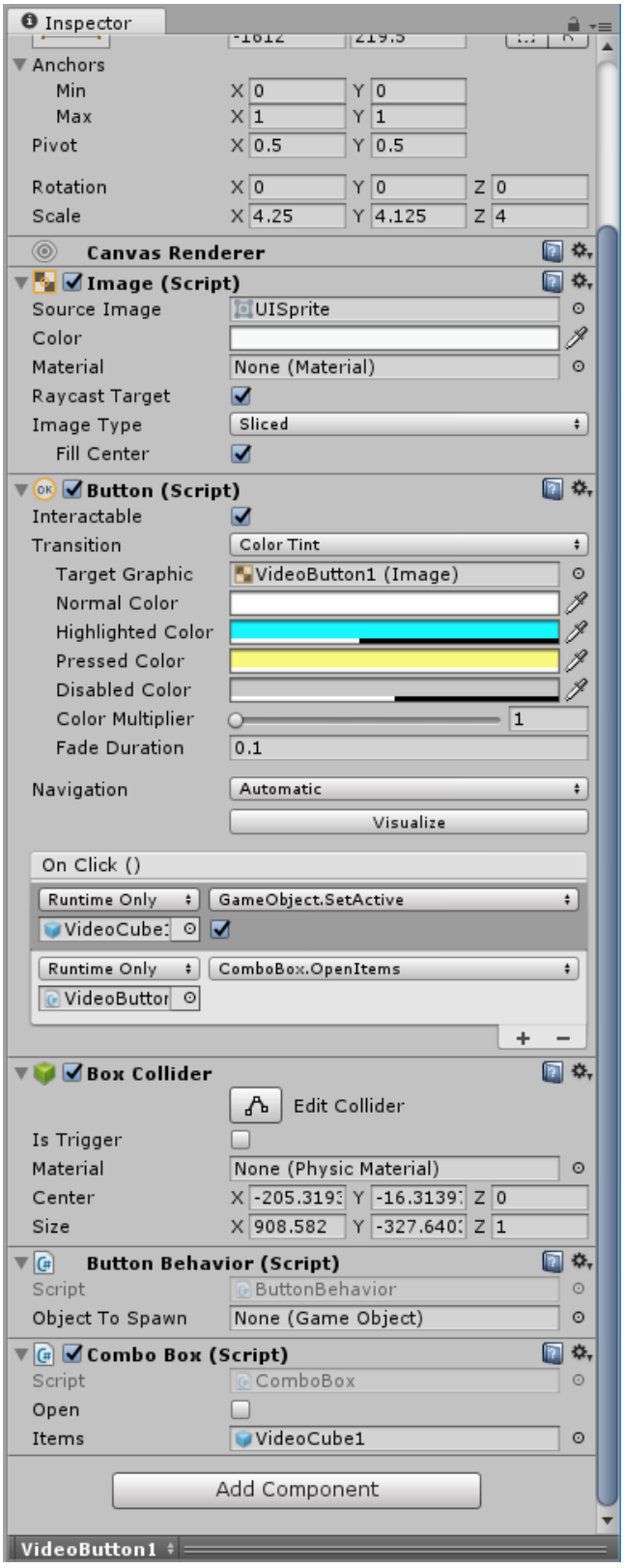


Fig. 3.48 Example of a button inspector

1. The prepared video should be saved as *.ogv* file and stored in the Unity path: `Assets/StreamingAssets/`.
2. Make a new material and set the texture as the video.
3. Write the C# script for making the video play. In this script, the Audio Source component is required so that it can show up on the Unity Inspector and give the sound from the video. The variables defined in the script is also shown on the Inspector. Since this project has more than one video to show in the different places, it is important to define a variable for choosing different videos. The script also need to tell Unity that if it has material to play.
4. Finally, go back to the Unity Inspect to set the options. The script *Video Player* and the material created for showing video have to be attached to the cube object.

- Clicker Input

Clicker Input is a GameObject with only one component (only C# script) to control the camera view on and off. All the details were written in the C# script. The *ClickerInput* class was designed to recognize the clicker event. The event callback can be configured in Unity Inspector.

- Load different scenes

The C# script *"LoadSceneOnClick"* was written for switching between different scenes. In this class, the function (*void LoadByIndex (int sceneIndex)*) passed the integer (given scene index) in from the button. When this function was called, the button would be able to load the desired scene.

- Moving object

The tracked object can move from point to point to indicate the user where to look. In this project, there are two moving objects, a right-hand shape 3D model, and a hand 2D image. The hand shape 3D model (see Figure 3.49) was designed in Autodesk® Maya (*.mb* file) and saved as *.fbx* file in order to import in Unity as a Prefab. A C# script was written to move the object. This script has to be attached to the object to make it move. The moving path was defined in another GameObject. The destination points are the children GameObjects of the path game object where the point position is given.

In the script which is called *pathfollower* class, *OnDrawGizmos()* is for the destination points visible in Unity scene. However, the Gizmo will not be seen in the application view

Fig. 3.49 3D model of hand shape

because Gizmo is only an editor function for visualizing in the scene view in Unity. Moving the object is done in the *Update()* function. The direction vector is calculated by subtracting the moving object position from the destination position. The position is updated according to the calculated vector.

- Texture material

In this project, some text information was needed to be given when the marker was detected. This text was placed on the surface of a 3D object, for example, a cube. When using 3D text from Unity directly, the 3D text appears on the top of the object but not on the surface. Following steps were followed to make the texture material:

1. Make a new shader called 3DText.
2. Make a new material and assign the Shader.
3. Assign the Font Texture on this material (see Figure 3.50).
4. Assign the material file to the Mesh Renderer portion on the GameObject.

Then the text can be on the surface of the 3D object as a material.

3.3.3 Information Presenting Design

The purpose of the application is to show the information regarding Ship Bridge. The text information was taken from the **Technical Manual Section 5b - Instrumentation**

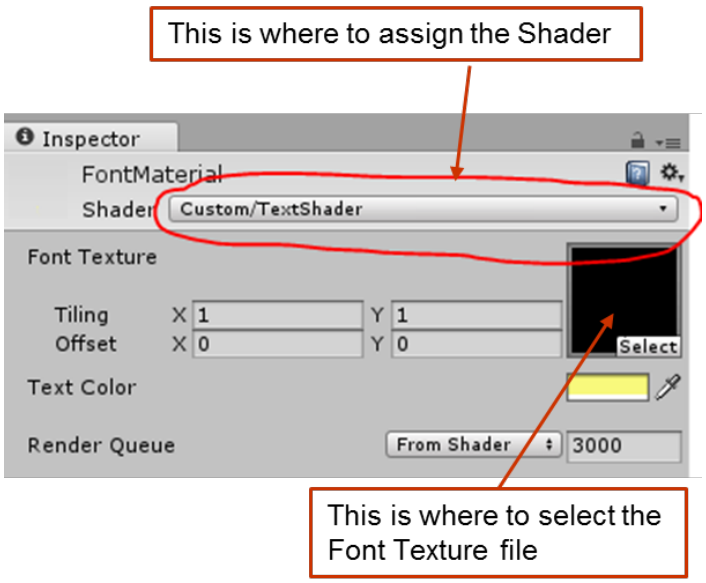


Fig. 3.50 Illustrate where to assign the Shader file and Font Texture file.

POLARIS [73]. Figure 3.51 shows the picture of Azimuth control on the Steering Panel. The video information was taken from the teacher and students from the Nautical and Navigation Team, Department of Engineering and Safety in UiT.

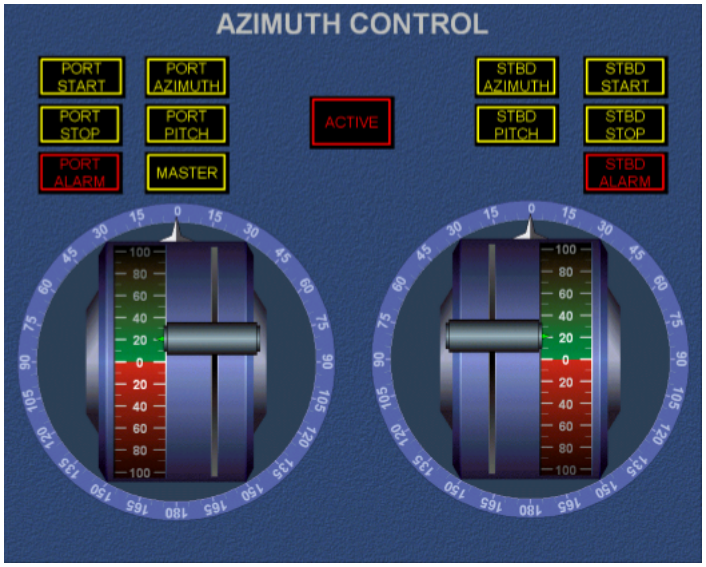


Fig. 3.51 Picture of Azimuth control on the Steering Panel from the manual[73].

Chapter 4

Results and Discussion

This chapter discusses the operation of the application (App). This App is a proof of concept that AR technology can be implemented for maritime training. The App was tried in the ship simulators (Kongsberg Ship Simulators, UiT The Arctic University of Norway). Results are obtained by using Mixed Reality Capture (Figure 4.1). The results from each scene are shown as follow. It should be noted that the captured scenes might not be as clear as they are in HoloLens.

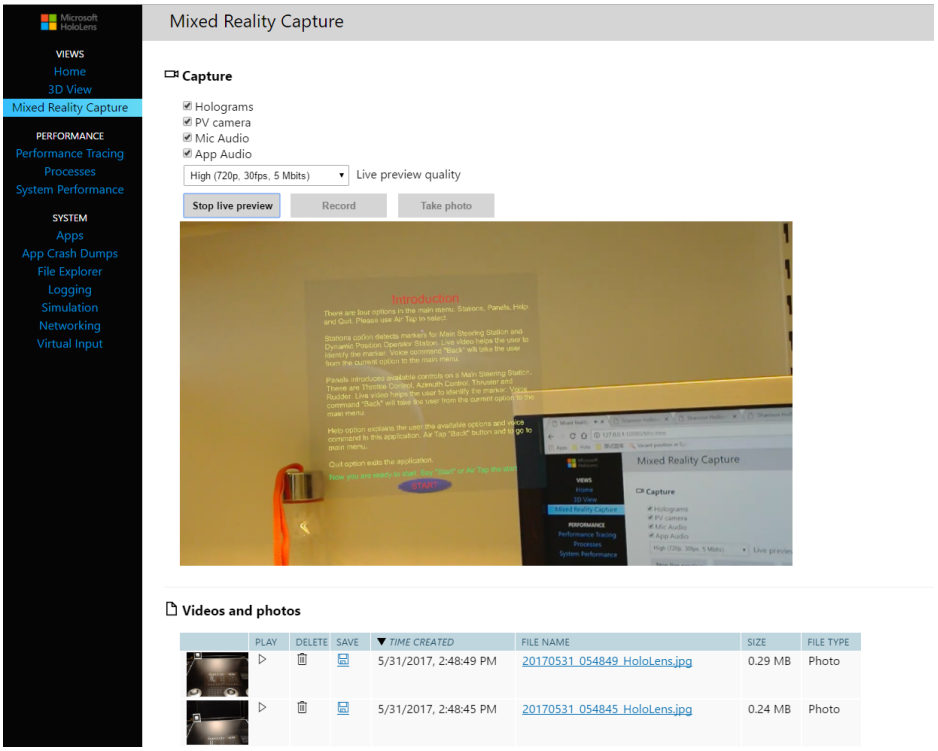


Fig. 4.1 Using Mixed Reality Capture to see the live preview.

4.1 Introduction

Once the App starts, the introduction scene shows up. Figure 4.2 shows the result in Unity scene, and Figure 4.3 shows the result in Ship Bridge environment.

Fig. 4.2 Introduction scene design result in Unity.

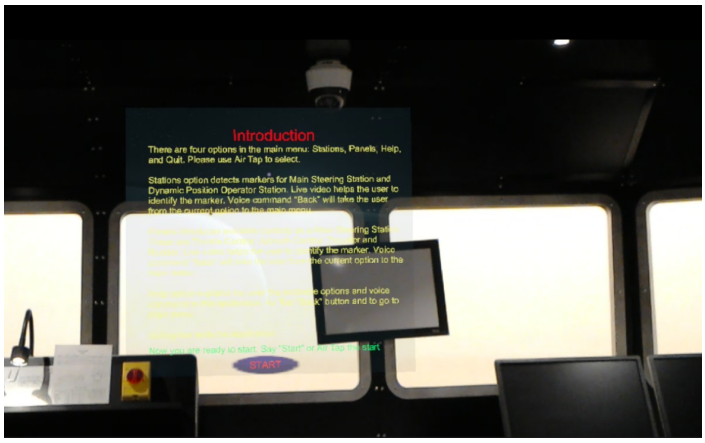


Fig. 4.3 Introduction scene running in Ship Bridge (Image is misguiding showing that text is difficult to read. However it was not the same when using HoloLens).

4.2 Main Menu Design

The App goes from introduction scene to the menu scene after voice command “start”. There are a title and four options on the menu. Help panel is also in this scene but not active. It is only active when the Help button is clicked. Figure 4.4 shows the menu scene in Unity. Figure 4.5 and Figure 4.6 shows the menu in the Ship Bridge environment. Figure 4.7 shows

the help panel in the Ship Bridge environment. It is seen that when the options are gazed, the button will change its colour to blue. When the button is selected, the button will change to green.

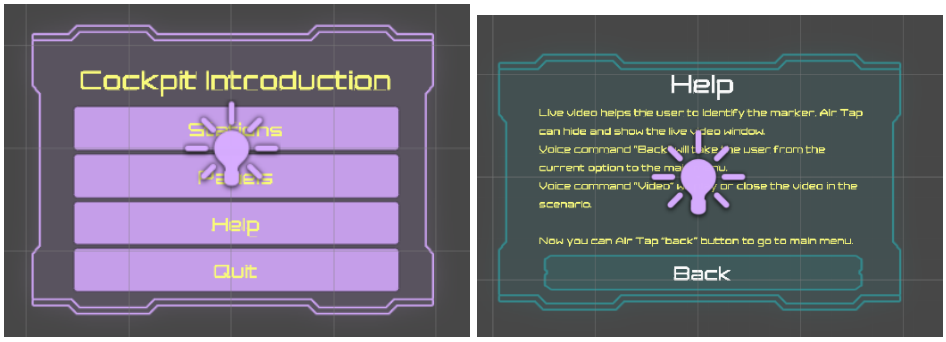


Fig. 4.4 Menu scene final aspect

Fig. 4.5 Menu in the Ship Bridge and the station option is gazed on.

4.3 Result of Station tracking scene design

Stations option has two marker tracking, one is for the Main Steering Station, and another one is for Dynamic Position Operator Station. Since Main Steering Station is where most operation happens, it will be given more information. Videos are shown in this tracking. Figure 4.8 and Figure 4.9 show the tracking result of Main Steering Station. Figure 4.10 shows the tracking result in Dynamic Position Operator Station.

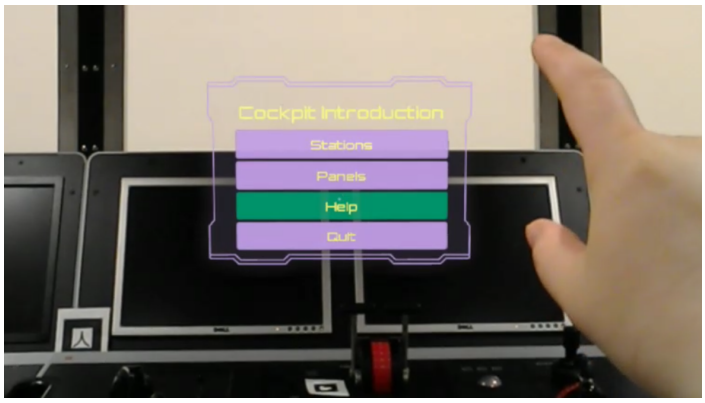


Fig. 4.6 Help button is selected.

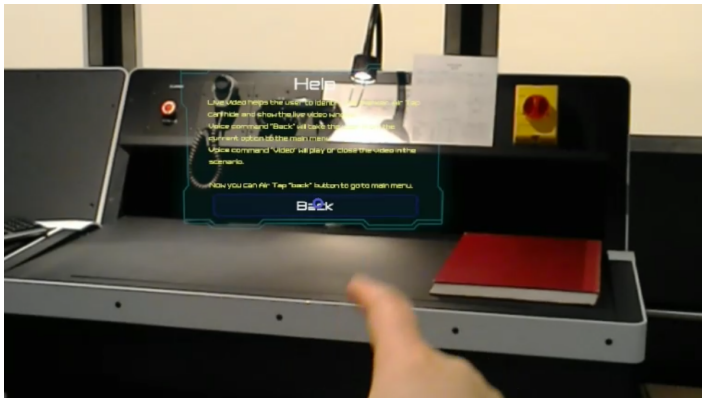


Fig. 4.7 Help panel in the Ship Bridge and back button is selected.

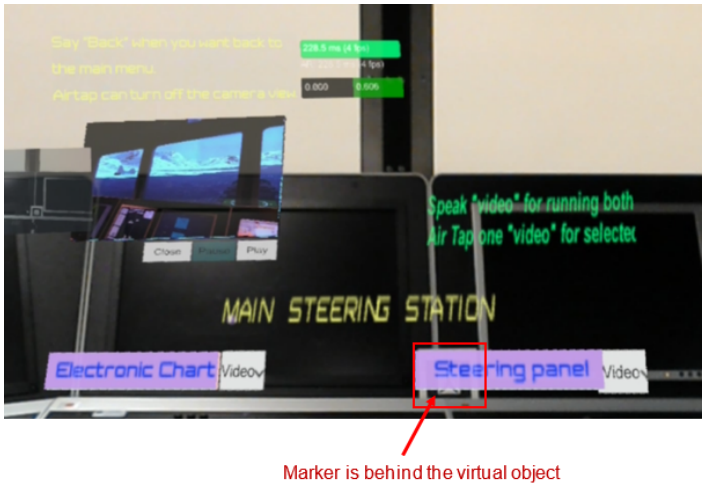


Fig. 4.8 Marker tracking on Steering Station

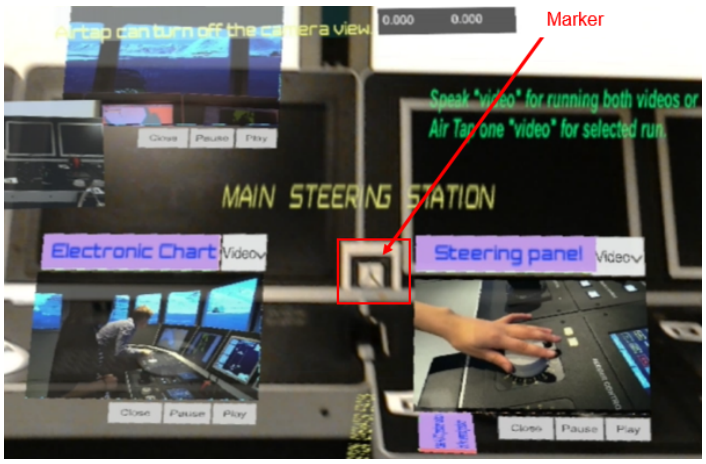


Fig. 4.9 Marker tracking on Steering Station when video option is active

Fig. 4.10 Information about Dynamic Position Operator Station

4.4 Panel Tracking Scene Design

WWith voice command “back”, the app will go back to the main menu. Air Tap *panel* option will go to the panel tracking scene. There are four markers on the panel to introduce the four controls: these are Throttle Control, Azimuth Control, Thruster, and Rudder. Figure 4.11 shows the position of the marker on the panel. From Figure 4.12 to Figure 4.15 show the tracking results of the four markers. All information is given by text, video and moving models.

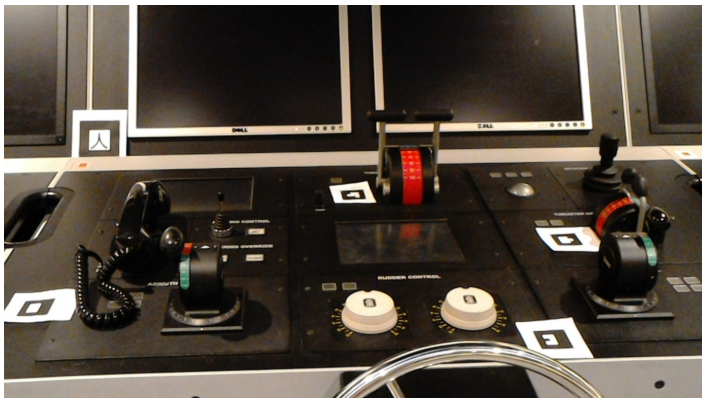


Fig. 4.11 Markers placed on the panel before the App start.

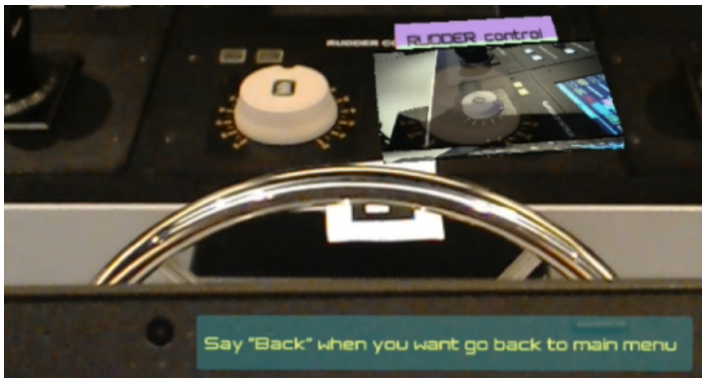
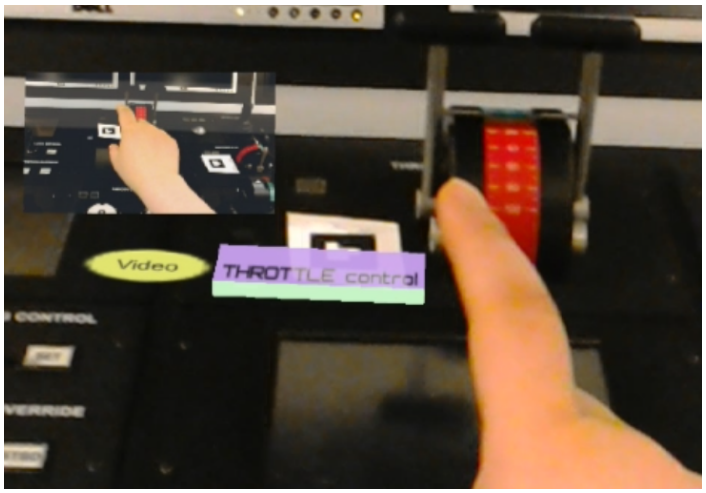


Fig. 4.12 Tracking result from Rudder control on the steering panel.

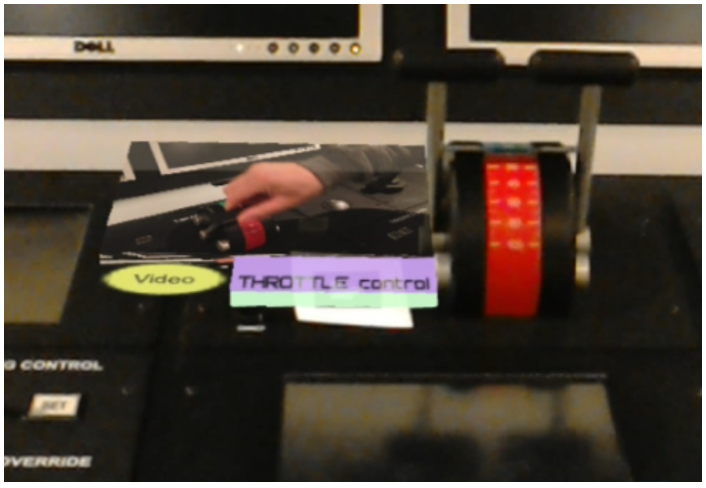
Most videos without voice introduction are shown up directly when the marker is detected. In tracking result of Throttle control button, the video shows up after voice command because this video contains voice introduction (Figure 4.13).

The moving object will show the operation. In Azimuth Control tracking, there is a 3D model hand guiding the operation (Figure 4.14). In Thruster control tracking, a 2D hand model is moving up and down for pointing the button (Figure 4.15).



(a)

(b)



(c)

Fig. 4.13 Tracking result from Throttle control button on the steering panel. (a) A live video shows the view from the HoloLens. The instruction Video is not shown at the beginning of the tracking. (b) The live video hides after the Air Tap. (c) After a voice command "video", video is on.

In order to give a better visualization of tracking, every tracking scene has a live video (for example in Figure 4.13(a)). Air Tap will control these live videos screen.

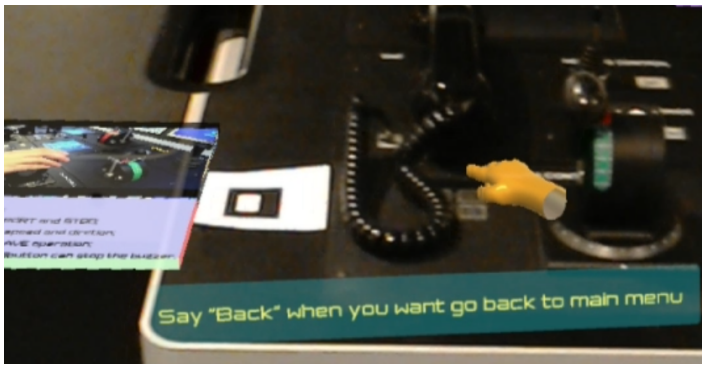


Fig. 4.14 Tracking results from Azimuth Control. The hand model shows the operation. It is moving from point to point according to the button on the panel.

Fig. 4.15 Tracking result from Thruster control button on the steering panel. Hand shape can move up and down to point the button.

4.5 Jitter Issue

The built app has jitter during tracking. This was due to the fact that HoloLens processing was being used at its max. It is possible to reduce it by either upgrading the hardware specs of the HoloLens (expected in future models) or optimizing the tracking algorithm. This issue is highlighted in the future works.

Efforts were made to reduce the jitter. It was found by trial and error that using 3D rather 2D texts, reduces the jitter. Therefore 3D texts were incorporated into all the virtual objects (as shown from 4.8 to Figure 4.15).

4.6 Summary

This AR App shows the basic information of a Ship Bridge. The App plays a role of tutor that can guide the trainees to know the environment of the Ship Bridge and basic operation of the panel on the station. When trainees are wearing the HoloLens, the information according to the Ship Bridge will be shown up in front of the trainees. The text information gives the detail of each position. The video information gives the knowledge to the trainees directly perceived through their senses. The moving objects tell the trainee the operation steps directly. The advanced features of the App such as voice command, gesture input, World anchor, etc. make the App user-friendly.

The markers used in this study are given in Appendix-A. They can be printed and used with the App.

The working of the App is recorded and provided as supplementary material. These can be seen from the following link:https://youtu.be/zzJ6mEH91F4

Chapter 5

Conclusions, Challenges, Limitations and Future Works

5.1 Conculsions

The objective of this thesis is to design an AR application for training maritime students. The developed app is only for a demonstration purpose and can be viewed as a proof of concept. The purpose of the app is to introduce a limited number of the stations and panels on a ship bridge. In this work, the Kongsberg Ship Bridge Simulator in UiT The Arctic University of Norway was focused. Microsoft® HoloLens has been chosen as a device for the app deployment. The app is built for Windows® 10 operating system (OS) and developed in Unity 5.5.1f1 Personal version.

- HoloLens was the good choice for training since it is a standalone device among all kinds of wearable AR devices. Discussed in Chapter 1.
- AR technology has been proven effective in training as discussed in Chapter 2.
- Marker-based tracking and markerless-based tracking were compared. It was found that marker based tracking is appropriate for the maritime training. Discussed in Chapter 3.
- In marker based tracking two different types of markers were considered. These are AR image markers and 2D-barcode markers. Discussed in Chapter 3.
- AR image markers are effective from long range however they take more processing to detect. Discussed in Chapter 3.

- 2D barcode markers are effective from a short distance and can be detected at the same time. Discussed in Chapter 3.
- In this study, different Standard Development Kits (ARToolkit and Vuforia® SDKs) were considered for tracking. ARToolkit SDK was used since its open source. However, Vuforia® SDK is not. Chapter 3.
- AR Application can be used for maritime training as demonstrated in Chapter 4.

5.1.1 Challenges

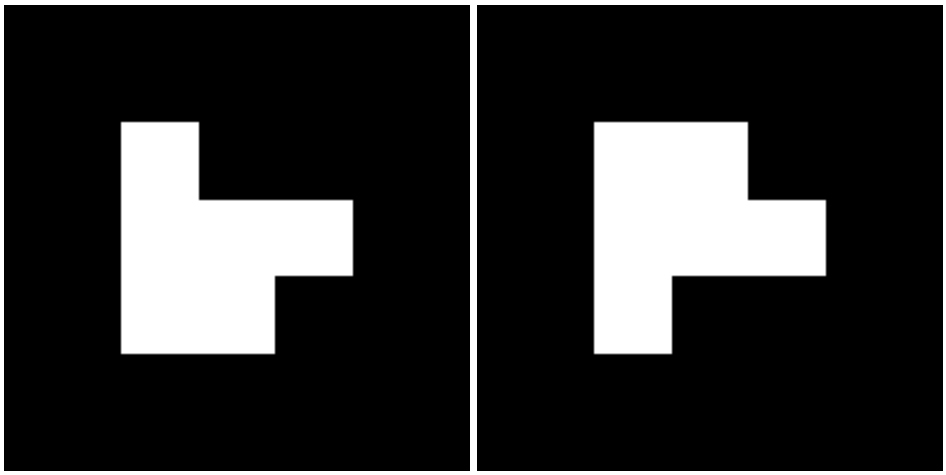
Following challenges were encountered during this project,

- It was my first attempt to work with an AR system. It was challenging in the sense that I had to learn multiple tools, SDKs, C# language, and Unity. Though it was hard and sometimes seemed impossible, but perseverance paid off. I managed to learn all and I produced a functional application. I am really proud of myself.
- Though in my earlier plans, I wanted to conduct the trials. These were cut short from my project because the application had jitter issues (discussed in section 4.5). I tried my best in available time to overcome the issue. It appears that the issue was more fundamental (HoloLens hardware limitation, tracking algorithm optimization) than could be fixed at this stage.

5.1.2 Limitations

The application have following limitations,

- The App only can run in the HoloLens. Emulator of HoloLens cannot run the App probably because the video configuration is fixed to 896×504.
- 2D-barcode marker ID does not match while tracking. The reason is that the image RGBA data for tracking is provided by Unity *WebcamTexture* object, which uses left-handed coordinate system. However, in ARToolKit, OpenCV, or OpenGL, the pixel coordinates are the right-handed coordinate system. This issue can be fixed by using mirrored IDs instead of original IDs while tracking and processing. Example for marker $ID = 1$ with mirrored $ID = 32$ is shown in 5.1.
- It is important that the users be aware that the information presentation is connected with the orientation of the marker. The user may be misguided if markers are placed in incorrectly.



(a) (b)

Fig. 5.1 2D-barcode marker (a) $ID = 1$ (b) $ID = 32$

5.1.3 Future Work

AR is still in an infancy stage. There are many problems to overcome and issues to explore in order to make AR widely used. Following are the list of future work,

- The trial is an important part of the App development. Questionnaires, interviews, and surveys can be done on the trials. Analyse the data and improve the App can be the next steps.
- Tracking stability can be improved. It was found that the developed App has jitter which could be reduced by the future models of HoloLens or by optimizing the tracking algorithm.
- For the purpose of introducing the Ship Bridge, the visual tracking was the only tracking method applied. In future work, sensor tracking and hybrid tracking can be used.
- Different Apps can be designed for the various purpose of maritime operations.
- More technology can be used for showing the maritime operation. For example, Leap Motion SDK can be used for tracking the teacher's operation by different sensors. The application will collect the sensors data which can be used to make a tutorial for the students.
- A different version of Unity has compatibility problems. An App designed in the previous version of Unity, might not be able to compile in the latest version of Unity.

This is not good to maintain an App for a long term basis. One of the solutions could be to choose another design tool for the App design such as Microsoft® DirectX.

References

[1] Graham M, Zook M, Boulton A. Augmented reality in urban places: contested content and the duplicity of code. Transactions of the Institute of British Geographers. 2013;38(3):464–479.

[2] Haller M. Emerging Technologies of Augmented Reality: Interfaces and Design: Interfaces and Design. Igi Global; 2006.

[3] Simsarian KT, Akesson KP. Windows on the World: An example of Augmented Virtuality;.

[4] Azuma RT. A survey of augmented reality. Presence: Teleoperators and virtual environments. 1997;6(4):355–385.

[5] Augmented Reality Glasses: What You Can Buy Now (or Soon);. Accessed date: 30.04.2017. Website. Available from: http://www.tomsguide.com/us/best-ar-glasses.

[6] A new way to see your world.;. Accessed date: 30.04.2017. Website. Available from: https://www.microsoft.com/en-us/hololens/hardware.

[7] Google unveils 'Project Glass' smart glasses. CNNMoney;. Accessed date: 01.05.2017. Website. Available from: http://money.cnn.com/2012/04/04/technology/google-project-glass/.

[8] Google 'Project Glass' Replaces the Smartphone With Glasses.;. Accessed date: 01.05.2017. Website. Available from: http://uk.pcmag.com/consumer-electronics-reviews-ratings/64927/news/google-project-glass-replaces-the-smartphone-with-glasses.

[9] SmartEyeglass;. Accessed date: 30.04.2017. Website. Available from: https://developer.sony.com/devices/mobile-accessories/smarteyeglass/.

[10] Lamkin P, a S Charara. The best smartglasses 2017 : Snap,Vuzix,ODG,Sony & more.;. Accessed date: 30.04.2017. Website. Available from: https://www.wareable.com/headgear/the-best-smartglasses-google-glass-and-the-rest.

[11] Donovan J. Epson Announces Moverio BT-300 Smart Glasses.;. Accessed date: 30.04.2017. Website. Available from: https://techcrunch.com/2016/02/22/epson-announces-moverio-bt-300-smart-glasses-at-mwc-2016/.

[12] What's Inside Google Glass? ;. Accessed date: 10.05.2017. Website. Available from: http://www.catwig.com/google-glass-teardown/.

[13] Moverio BT-300FPV Smart Glasses (FPV/Drone Edition);. Accessed date: 10.05.2017. Website. Available from: https://epson.com/For-Home/Wearables/Smart-Glasses/Moverio-BT-300FPV-Smart-Glasses-(FPV-Drone-Edition)-/p/V11H756020F.

[14] Minagawa J, Choi W, Tsurumi S, Hirakoso N, Sasaki N, Li L, et al. Supporting system with hand gesture for location-based augmented reality. In: Consumer Electronics (GCCE), 2015 IEEE 4th Global Conference on. IEEE; 2015. p. 479–480.

[15] Nango J. The bridge simulator.;. Accessed date: 10.05.2017. Website. Available from: https://publicartnorway.org/prosjekter/uit-the-arctic-university-of-norway-new-technology-building/.

[16] Kirner C, Cerqueira CS, Kirner TG. Using augmented reality cognitive artifacts in education and virtual rehabilitation. In: Virtual Reality in Psychological, Medical and Pedagogical Applications. InTech; 2012. .

[17] Carmigniani J, Furht B, Anisetti M, Ceravolo P, Damiani E, Ivkovic M. Augmented reality technologies, systems and applications. Multimedia Tools and Applications. 2011;51(1):341–377.

[18] Henderson SJ, Feiner SK. Augmented reality for maintenance and repair (armar). DTIC Document; 2007.

[19] Ternier S, Klemke R, Kalz M, Van Ulzen P, Specht M. ARLearn: Augmented Reality Meets Augmented Virtuality. J UCS. 2012;18(15):2143–2164.

[20] Lee K. Augmented reality in education and training. TechTrends. 2012;56(2):13–21.

[21] Schoenfeld A. Learning to think mathematically: Problem solving, metacognition, and sense-making in mathematics . Coleccion Digital Eudoxus. 2009;(7).

[22] Augmented Reality Education Apps.;. Accessed date: 29.05.2017. Website. Available from: https://thinkmobiles.com/augmented-reality-education/.

[23] ;. Accessed date: 28.05.2017. Website. Available from: http://www.digitalavmagazine.com/wp-content/uploads/2014/01/Zientia-1.jpg.

[24] Biology: Organs;. Accessed date: 29.05.2017. Website. Available from: http://www.arlearning.co.uk/#demo.

[25] The User Interface from Front to Back (Augmented Reality);. Accessed date: 29.05.2017. Website. Available from: http://csis.pace.edu/~marchese/DPS/Lect3/dpsl3.html.

[26] Bichlmeier C, Wimmer F, Heining SM, Navab N. Contextual anatomic mimesis hybrid in-situ visualization method for improving multi-sensory depth perception in medical augmented reality. In: Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on. IEEE; 2007. p. 129–138.

[27] Google glass for war: The US military funded smart helmet that can beam information to soldiers on the battlefield.;. Accessed date: 29.05.2017. Website. Available from: http://www.dailymail.co.uk/sciencetech/article-2640869/Google-glass-war-US-military-reveals-augmented-reality-soldiers.html.

[28] Richardson T, Gilbert S, Holub J, Thompson F, MacAllister A, Radkowski R, et al. Fusing Self-Reported and Sensor Data from Mixed-Reality Training. In: The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC); 2014. .

[29] Henderson SJ, Feiner S. Evaluating the benefits of augmented reality for task localization in maintenance of an armored personnel carrier turret. In: Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on. IEEE; 2009. p. 135–144.

[30] Hyundai's augmented reality manual: A simple but extremely good idea.;. Accessed date: 03.03.2017. Website. Available from: https://arstechnica.com/cars/2016/01/hyundais-augmented-reality-manual-a-simple-but-extremely-good-idea/.

[31] Hololens Plant Maintenance.;. Accessed date: 29.05.2017. Website. Available from: http://https://www.youtube.com/watch?v=QTuKcm8s4QQ.

[32] Big Stock by shutterstock;. Accessed date: 29.05.2017. Website. Available from: https://www.bigstockphoto.com/image-159586733/.

[33] Pokemon Go – 3 Aspects for Parents to Consider.;. Accessed date: 29.05.2017. Website. Available from: http://kidsinministry.org/pokemon-go-3-aspects-parents-consider/.

[34] Schröder-Hinrichs JU, Hollnagel E, Baldauf M, Hofmann S, Kataria A. Maritime human factors and IMO policy. Maritime Policy & Management. 2013;40(3):243–260.

[35] Jaeyong O, Park S, Kwon OS. Advanced Navigation Aids System based on Augmented Reality. International Journal of e-Navigation and Maritime Economy. 2016;5:21–31.

[36] Procee S, Baldauf M. Augmented Reality as part of a man-machine interface in e-Navigation. ResearchGate;. Available from: https://www.researchgate.net/profile/Michael_Baldauf/publication/273057248_Augmented_Reality_in_Ships_Bridge_Operation/links/56a29c5e08aeef24c585f98f.pdf/.

[37] BUILD 2015: A closer look at the Microsoft HoloLens hardware.;. Accessed date: 22.05.2017. Website. Available from: https://blogs.windows.com/devices/2015/04/30/build-2015-a-closer-look-at-the-microsoft-hololens-hardware/#IFiobsAyqhDG8JIO.97.

[38] Use gesture;. Accessed date: 22.05.2017. Website. Available from: https://support.microsoft.com/en-us/help/12644/hololens-use-gestures.

[39] Kushwaha V. Microsoft HoloLens; 2016. Accessed date: 22.05.2017. Website. Available from: http://technewsexpres.blogspot.no/2016/03/microsoft-hololens.html.

[40] HoloLens hardware details.;. Accessed date: 22.05.2017. Website. Available from: https://developer.microsoft.com/en-us/windows/mixed-reality/hololens_hardware_details.

[41] HoloLens - Introduction to the HoloLens, Part 2: Spatial Mapping.;. Accessed date: 02.05.2017. Website. Available from: https://msdn.microsoft.com/en-us/magazine/mt745096.aspx.

[42] Microsoft HoloLens Official Site;. Accessed date: 22.05.2017. Website. Available from: https://www.microsoft.com/en-us/hololens?WT.mc_id=Search_EN_ModD_2_Hololens_030216,.

[43] Install the tools;. Accessed date: 22.05.2017. Website. Available from: https://developer.microsoft.com/en-us/windows/mixed-reality/install_the_tools,.

[44] Petit A. Robust model-based tracking for markerless complex objects.;. Accessed date: 20.05.2017. Website. Available from: https://www.youtube.com/watch?v=PtdxNAvso08.

[45] Garon M, Boulet P, Doironz JP, Beaulieu L, Lalonde JF. Real-Time High Resolution 3D Data on the HoloLens. In: Mixed and Augmented Reality (ISMAR-Adjunct), 2016 IEEE International Symposium on. IEEE; 2016. p. 189–191.

[46] Weng ENG, Khan RU, Adruce SAZ, Bee OY. Objects tracking from natural features in mobile augmented reality. Procedia-Social and Behavioral Sciences. 2013;97:753–760.

[47] Natural Features and Image Ratings.;. Accessed date: 20.05.2017. Website. Available from: https://library.vuforia.com/articles/Solution/Natural-Features-and-Ratings.

[48] Saunier N, Sayed T. A feature-based tracking algorithm for vehicles in intersections. In: Computer and Robot Vision, 2006. The 3rd Canadian Conference on. IEEE; 2006. p. 59–59.

[49] Corné, T., Marker Detection for AR Applications.;. Accessed date: 20.05.2017. Website. Available from: https://infi.nl/nieuws/marker-detection-for-augmented-reality-applications/.

[50] Celozzi C, Paravati G, Sanna A, Lamberti F. A 6-DOF ARTag-based tracking system. IEEE Transactions on Consumer Electronics. 2010;56(1).

[51] Jun J, Yue Q, Qing Z. An extended marker-based tracking system for augmented reality. In: Modeling, Simulation and Visualization Methods (WMSVM), 2010 Second International Conference on. IEEE; 2010. p. 94–97.

[52] Santos PC, Stork A, Buaes A, Pereira CE, Jorge J. A real-time low-cost marker-based multiple camera tracking solution for virtual reality applications. Journal of Real-Time Image Processing. 2010;5(2):121–128.

[53] Köhler J, Pagani A, Stricker D. Detection and identification techniques for markers used in computer vision. In: OASIcs-OpenAccess Series in Informatics. vol. 19. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik; 2011. .

[54] SEARCHHIGH L. Theory and applications of marker-based augmented reality;.

[55] Amin D, Govilkar S. Comparative study of augmented reality sdk's. International Journal on Computational Science & Applications. 2015;5(1):11–26.

[56] Naimark L, Foxlin E. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In: Mixed and Augmented Reality, 2002. ISMAR 2002. Proceedings. International Symposium on. IEEE; 2002. p. 27–36.

[57] Augmented Reality als Game Anwendung;. Accessed date: 20.05.2017. Website. Available from: http://winfwiki.wi-fom.de/index.php/Augmented_Reality_als_Game_Anwendung.

[58] VuMark;. Accessed date: 20.05.2017. Website. Available from: https://library.vuforia.com/articles/Training/VuMark.

[59] Best AR SDK for Industry vertical AR solutions: Paid (Vuforia/Wikitude) or Open Source (ARToolkit)?;. Accessed date: 20.05.2017. Website. Available from: https://www.linkedin.com/pulse/best-ar-sdk-industry-vertical-solutions-paid-open-source-prabhu?trk=prof-post.

[60] Qian L, Azimi E, Kazanzides P, Navab N. Comprehensive Tracker Based Display Calibration for Holographic Optical See-Through Head-Mounted Display. arXiv preprint arXiv:170305834. 2017;.

[61] Walkthrough: Creating and Using a Dynamic Link Library (C++);. Accessed date: 12.04.2017. Website. Available from: https://msdn.microsoft.com/en-us/library/ms235636(v=vs.100).aspx?ranMID=24542&ranEAID=TnL5HPStwNw&ranSiteID=TnL5HPStwNw-cdYAXa4oVwWbEfOCYl9VHA&tduid=(17005bc7d93c8708e4d0a32af2089638)(256380)(2459594)(TnL5HPStwNw-cdYAXa4oVwWbEfOCYl9VHA)().

[62] Use C++ codes in a C# project — unmanaged C++ solution;. Accessed date: 12.04.2017. Website. Available from: https://drthitirat.wordpress.com/2013/05/30/combine-gui-of-c-with-c-codes/.

[63] Locatable camera;. Accessed date: 20.05.2017. Website. Available from: https://https://developer.microsoft.com/en-us/windows/mixed-reality/locatable_camera.

[64] Tutorial 3 : Matrices;. Accessed date: 20.05.2017. Website. Available from: http://www.opengl-tutorial.org/beginners-tutorials/tutorial-3-matrices/#fn:projection.

[65] Zhang X, Fronz S, Navab N. Visual marker detection and decoding in AR systems: A comparative study. In: Proceedings of the 1st International Symposium on Mixed and Augmented Reality. IEEE Computer Society; 2002. p. 97.

[66] Zhang Z. A flexible new technique for camera calibration. IEEE Transactions on pattern analysis and machine intelligence. 2000;22(11):1330–1334.

[67] Hartley RI, Zisserman A. Multiple View Geometry in Computer Vision. 2nd ed. Cambridge University Press, ISBN: 0521540518; 2004.

[68] Tuceryan M, Greer DS, Whitaker RT, Breen DE, Crampton C, Rose E, et al. Calibration requirements and procedures for a monitor-based augmented reality system. IEEE Transactions on Visualization and Computer Graphics. 1995;1(3):255–273.

[69] Camera calibration With OpenCV;. Accessed date: 20.03.2017. Website. Available from: http://docs.opencv.org/2.4/doc/tutorials/calib3d/camera_calibration/camera_calibration.html.

[70] Khan D, Ullah S, Rabbi I. Factors affecting the design and tracking of ARToolKit markers. Computer Standards & Interfaces. 2015;41:56–66.

[71] Barcode marker generator.;. Accessed date: 25.02.2017. Website. Available from: http://www.artoolworks.com/support/applications/marker/.

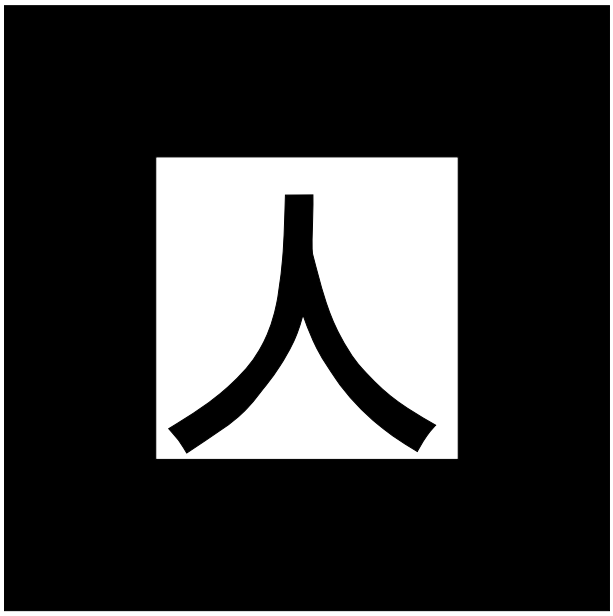
[72] The Unity Editor Basics.;. Accessed date: 28.03.2017. Website. Available from: https://hololens.reality.news/how-to/hololens-dev-101-unity-editor-basics-0175161/.

[73] (s) PS. Technical Manual Section 5b - Instrumentation POLARIS. Kongsberg Maritime; 2014.

Appendix A

Marker patterns

AR markers used for Main Steering Station tracking

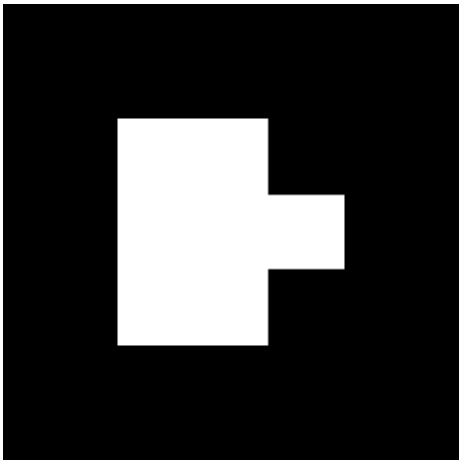


AR markers used for Dynamic Operation Station tracking

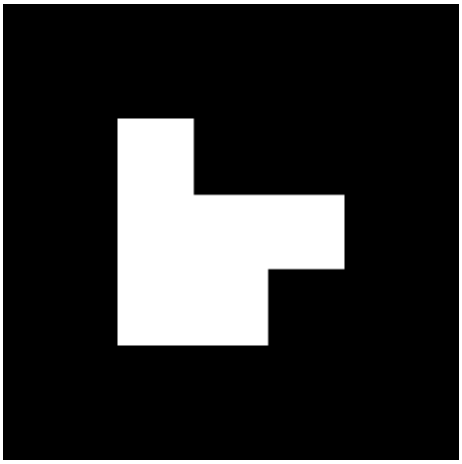
2D barcode markers

For Rudder tracking

For Thruster Control tracking



For Throttle Control tracking



For Azimuth Control tracking