Aalto University
School of Science
Master’s Programme in ICT Innovation

Wenbin Luo

Building An AR Treasure Hunt Game Using Indoor Localization Technology

Master’s Thesis
Espoo, September 5th, 2018

Supervisor: Prof. Petri Vuorimaa
Advisor: D.Sc. Jiang Dong
Augment reality is currently a popular technology, which can insert and overlay digital and virtual information into the real world. Meanwhile, indoor positioning is another technique gaining a lot of attentions in its field. AR games with GPS outdoor positioning have been very successful recently, one great example of them is Pokémon Go.

As a company offering indoor positioning services, VIMAI is committed to exploring the application of indoor positioning. One of their interesting fields is the feasibility of combining indoor positioning with augmented reality. However, no one has researched the combination of AR and indoor positioning.

This thesis covers the research of the two technologies, as well as the design and development of an AR game with indoor localization. In the research part, the elementary theory of these technologies were studied. In the implementation part, a server-client game system was developed, including an AR treasure hunt game client and a data storage server. The client displays the treasure using AR and register the user using image-based indoor locating method. The server stores game data through RESTful requests. In the evaluation part, the game by doing a user test was evaluated. The feedback and comments of the game were collected and discussed in the end.

### Keywords
- Augment Reality
- Indoor localization
- RESTful API
- Unity
- Vuforia

### Language
- English
Acknowledgements

First, I would like to thank my supervisor, Petri Vuorimaa, who provided his expertise and patience that helped me so much during the thesis research and writing. Then, I would like to thank my adviser, Jiang Dong, who gave me guidance and advice during I had my internship at VimAI. Thanks to VimAI for giving me this opportunity to work on the thesis project. I will remember the happy time with Marius, Vilen, and Ismo, working at VimAI is an unforgettable experience for me. Wish you guys all the best!

Finally, I would send my best regard to my parents and my girlfriend, who support me all the time during I am studying aboard.

Espoo, Finland September 5th, 2018

Wenbin Luo
### Abbreviations and Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR</td>
<td>Augmented Reality</td>
</tr>
<tr>
<td>LBS</td>
<td>Location Based Service</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>IPS</td>
<td>Indoor Positioning System</td>
</tr>
<tr>
<td>HMD</td>
<td>Head-Mounted Display</td>
</tr>
<tr>
<td>LFU</td>
<td>Low-Frequency Ultrasound</td>
</tr>
<tr>
<td>VSLAM</td>
<td>Visual Simultaneous Localization and Mapping</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Development Kit</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio-Frequency Identification</td>
</tr>
<tr>
<td>PDR</td>
<td>Pedestrian Dead-Reckoning</td>
</tr>
<tr>
<td>RSSI</td>
<td>Received Signal Strength Indication</td>
</tr>
<tr>
<td>SfM</td>
<td>Structure from Motion</td>
</tr>
<tr>
<td>QR code</td>
<td>Quick Response code</td>
</tr>
<tr>
<td>SIFT</td>
<td>Scale-Invariant Feature Transform</td>
</tr>
<tr>
<td>SM</td>
<td>Structure from Motion</td>
</tr>
<tr>
<td>RPC</td>
<td>Remote Procedure Calls</td>
</tr>
<tr>
<td>RANSAC</td>
<td>Random Sample Consensus</td>
</tr>
</tbody>
</table>
## Contents

Abbreviations and Acronyms ........................................... 4

1 Introduction ......................................................... 7
   1.1 Problem Statement ......................................... 7
   1.2 Structure of the Thesis ................................... 8

2 Background ........................................................... 9
   2.1 Augmented Reality Overview .............................. 9
   2.2 AR System Components .................................... 11
      2.2.1 Mobile Computing Platform ....................... 11
      2.2.2 Software Framework ................................. 11
      2.2.3 Display Part .......................................... 12
   2.3 AR Registration ........................................... 14
      2.3.1 Sensor-based Methods ............................... 15
      2.3.2 Vision-based Methods ............................... 16
   2.4 AR Applications ........................................... 19
      2.4.1 Tourism and Navigation ............................ 19
      2.4.2 Training and Education ............................ 20
      2.4.3 Assembly and Maintenance ....................... 20
      2.4.4 Advertisement and Game ........................... 21
   2.5 Indoor Localization ....................................... 23
      2.5.1 Localization Methods ............................... 23
      2.5.2 Labeling .............................................. 23
      2.5.3 Triangulation ........................................ 23
      2.5.4 Dead reckoning ...................................... 24
      2.5.5 Fingerprinting ....................................... 24
      2.5.6 Image-based .......................................... 24

3 System Design and Environment ................................ 26
   3.1 System Design ............................................. 26
      3.1.1 Localization Method ................................. 26
<table>
<thead>
<tr>
<th>Sections</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.2 System Requirement</td>
<td>27</td>
</tr>
<tr>
<td>3.1.3 Example Game Scenario</td>
<td>28</td>
</tr>
<tr>
<td>3.2 System Architecture</td>
<td>28</td>
</tr>
<tr>
<td>3.2.1 Client Architecture</td>
<td>29</td>
</tr>
<tr>
<td>3.2.2 Server Architecture</td>
<td>30</td>
</tr>
<tr>
<td>3.3 Software Environment</td>
<td>31</td>
</tr>
<tr>
<td>3.3.1 Unity Components</td>
<td>32</td>
</tr>
<tr>
<td>3.4 Hardware Environment</td>
<td>33</td>
</tr>
<tr>
<td>4 Implementation</td>
<td>34</td>
</tr>
<tr>
<td>4.1 Image Data Modification</td>
<td>34</td>
</tr>
<tr>
<td>4.1.1 Image Compression</td>
<td>35</td>
</tr>
<tr>
<td>4.2 Position Data Server</td>
<td>37</td>
</tr>
<tr>
<td>4.3 Scene Control</td>
<td>38</td>
</tr>
<tr>
<td>4.4 Calculating the Distance</td>
<td>38</td>
</tr>
<tr>
<td>4.4.1 Euclidean Distance</td>
<td>38</td>
</tr>
<tr>
<td>4.4.2 Threshold D</td>
<td>39</td>
</tr>
<tr>
<td>4.4.3 Visualization</td>
<td>40</td>
</tr>
<tr>
<td>4.5 Vuforia World Center Mode</td>
<td>41</td>
</tr>
<tr>
<td>4.6 Multiplayer Mode</td>
<td>42</td>
</tr>
<tr>
<td>4.7 Treasure</td>
<td>45</td>
</tr>
<tr>
<td>4.7.1 Treasure Placement</td>
<td>45</td>
</tr>
<tr>
<td>4.7.2 Capture Treasure</td>
<td>45</td>
</tr>
<tr>
<td>5 Evaluation</td>
<td>47</td>
</tr>
<tr>
<td>5.1 Result Evaluation</td>
<td>47</td>
</tr>
<tr>
<td>5.1.1 Performance</td>
<td>48</td>
</tr>
<tr>
<td>5.2 User Test</td>
<td>52</td>
</tr>
<tr>
<td>6 Conclusions</td>
<td>55</td>
</tr>
<tr>
<td>6.1 Future Work</td>
<td>56</td>
</tr>
<tr>
<td>6.2 Final Thought</td>
<td>57</td>
</tr>
<tr>
<td>A Response Data Format</td>
<td>63</td>
</tr>
<tr>
<td>B Questionnaire</td>
<td>64</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Augmented reality (AR) is an emerging technology, which overlays computer-generated virtual contents over the real world. Due to the development of open source AR code recently and the advent of small powerful and highly integrated devices such as smartphones, AR glasses, the AR market is really opening up with developers increasingly searching for ways to make us everyday a lot more exciting. The number of AR applications has increased rapidly in the past few years. AR can be applied to various real-life scenarios, such as navigation, education, training, maintenance, and entertainment. A great example of them is Pokémon Go, one of the most popular game in 2016, which had millions of daily active users and earned millions of dollars at its time. It is a Location-Based Service (LBS) game, which uses Global Positioning System (GPS) to locate and capture virtual creatures, which appear as if they are in the player’s real-world location by AR. This is an obvious example that AR has a positive impact on mobile applications.

In the meantime, indoor positioning technology is also gaining a lot of attention from the research field. Indoor localization uses sensory data or image input to achieve the localization. Before indoor positioning is widely used, LBS always used GPS to get the geolocation data of mobile devices. However, GPS does not work well in indoor environment, because the signal can be attenuated by the roofs and walls. The development of indoor positioning technology fills the gap in this field.

1.1 Problem Statement

The success of Pokémon Go indicates the potential of building games with AR and LBS. However, the limitation of GPS restricts the application of indoor AR. No one has developed a geolocation-based AR game using indoor
localization technology, yet. Hence, researches in this field remain to be carried out.

VimAI Oy is a company offering advanced indoor positioning service. They use image-based method to localize objects and people in indoor space with high accuracy. The company is committed to exploring the application of indoor positioning technology. Currently, they provide services including indoor navigation, object detection, and several other indoor position related services. They attempt to discover the feasibility of combining indoor localization with AR and how indoor positioning would exist in a treasure hunt game like Pokémon Go. This is the origin of the idea of this thesis.

The goal of this thesis is to examine the current technologies of AR and indoor positioning. As well as to design and develop an AR treasure hunt game using indoor positioning technology, which can support most of the mobile devices. During working on the thesis, I attempt to answer the following research questions:

- How can we combine the existing AR and indoor positioning technologies?
- What are the special problem areas when using indoor positioning in AR?

1.2 Structure of the Thesis

The rest part of this thesis is structured as following: Chapter 2 provides background knowledge about AR and indoor positioning from self study and literature review. Chapter 3 discusses the design of the game method and system architecture, as well as the developing environment. Chapter 4 focuses on the implementation of the game, including encountered problems and corresponding solutions. Chapter 5 evaluates the output implementation of the thesis. Chapter 6 addresses the conclusion of the thesis and talks about the future work.
Chapter 2

Background

This chapter introduces the technologies and theories that were studied during the thesis in a nutshell. Section 2.1 gives the overview of augmented reality, which covers its definition, history and different types. Section 2.2 describes the AR system components. In section 2.3, registration and tracking theories of AR are introduced. Section 2.4 provides existing applications of AR. In the following part, the summary of indoor positioning techniques are reviewed in section 2.5. A special attention is put on image-based localization, since the system uses this method for indoor positioning in this thesis.

2.1 Augmented Reality Overview

Augmented reality is an enhanced version of the physical, real-world reality of which elements are superimposed by computer-generated or extracted real-world sensory input such as sound, video, graphics or haptics. [30] It works as an enhancement of the real world with virtual objects in it, when a person’s real environment is supplemented or augmented with computer generated images.

Although the word “augmented reality” has been created since 1992 by Thomas Caudell, a researcher from Boeing [21], AR just starts becoming a popular topic in recent years. The whole industry is benefiting from the wide availability of consumer electronics with high-performance processors and high-accuracy accelerators. As the technology is growing more and more developed, thousands of commercial and industrial solutions are adopted using AR nowadays, which makes AR a big marketplace that many IT companies are focusing on. As the development of high performance devices and excellent cross-platform SDKs, AR can be implemented and displayed on
various of platforms. From affordable smartphones and tablets to expensive
dedicated headset and helmet. Most newly released smartphones support
primary AR functions and some flagship cellphones provide space under-
standing ability, like Google Pixel, Samsung Galaxy S8 and iPhone X. In the
meantime, many tech companies launched their own dedicated AR devices,
for example, Google released Google Glass in 2014 and Microsoft released a
mixed reality smart glasses HoloLens in 2016. In the future, we may expect
AR can work on every personal device and integrate the daily life of people
more than it does now.

The basic concept behind an AR interface is to analyze the current envi-
ronment in order to project images or 3D objects onto it ideally, and to keep
the user connected to the virtual world without breaking the real world. In
order to connect the real world with the virtual world, two types of AR are
popularly used, which are marker-based AR and markerless AR.

Marker-based AR is probably the most common ways of AR currently. It
uses predetermined patterns as the marker, which could be a QR code or an
image, and stores them in a database. Then, the application uses computer
vision algorithm to look for the pattern in the environment. The theory
behind that is finding the feature points in the environment and checking
if the captured feature points match those of the marker. As long as it
recognizes the pattern, the virtual 3D object would be displayed onto that.
In marker-based AR, the marker is the bridge between real world and virtual
world. A good marker should be easily detected in every situation so that
the application is aware of the position and scale of real world reference.
In Figure 2.1, a cellphone-generated 3D low poly Trump model is being
displayed on the dollar. In this case, the dollar works as the marker. It
seems that the model just stands on the dollar paper with perfect adherence.

Markerless AR, unlike marker-based AR, does not reference predefined
patterns in the database. It means the system does not need to acquire the
pre-knowledge of the surrounding environment. On the contrary, the system
looks for existing identifiers or other features in the environment, which could
be a ground, an arbitrary cubic object, or a specific geography location. Figure
2.2 shows a laptop, which is created by markerless AR. Markerless AR
becomes more frequent nowadays due to the broad availability of inertial sen-
sors, GPS, and accelerometer of mobile phones. Details about technologies
of marker-based and markerless AR will be discussed in section 2.3.
2.2 AR System Components

This section introduces fundamental parts of an ordinary AR system. A typical AR system should have functions including registration, tracking, virtual object rendering, and displaying. Dimitris et al. [7] introduced three essential parts that are always required for a fully functional system, which are mobile computing platform, software framework, and display part.

2.2.1 Mobile Computing Platform

AR systems should allow users to move freely in the real environment, and thus a mobile platform is required to liberate the user from the restraint of wired equipment. A large and heavy device is impractical for mobile usage. Meanwhile, the mobile platform needs to be efficient and fast in computing in order to run the software and calculate the pose and position rapidly. Thus, a mobile computing platform is the applicable choice to run an AR system. Many popular platforms nowadays could run AR in excellent performance including tablets, mobile phones, and wearable glasses. These platforms have made remarkable progress recently in every facet from embedded inertial sensors to advanced cameras.

2.2.2 Software Framework

It is possible to completely write the code from scratch, but it would be an incredible tedious and time-consuming process for developers to do this every time they want to build a software. As a consequence, a framework or SDK is very important for AR development. When people develop high-level AR applications, the low-level implementation should not bother the developer.
SDK stands for Software Development Kit, it usually contains a development environment, which allows the user to write code and use predefined interfaces and methods, and other developing tools for debugging, testing and building. Today many organizations and companies have already developed SDKs. In the following, some representative examples are reviewed.

- **ARCore**: ARCore is developed by Google, which mainly focuses on Android development. ARCore has three key capabilities, that are motion tracking, environmental understanding and light estimation. Motion tracking allows the phone get the information of its own posture and position related to the real world. Environment understanding enables the phone to detect and recognize surfaces in the real world, such as dining tables or walking floors. Light estimation provides the phone with the ability to detect the lighting condition around itself. These capabilities given by ARCore makes building sophisticated AR applications possible. In the future, ARCore will become one of the leading SDKs in Android platform.

- **ARKit**: ARKit is an AR SDK made by Apple for iOS development. This framework offers quite similar functions to ARCore, including motion tracking, environmental understanding and light estimation. ARKit only supports iOS, but currently many apps chooses ARKit as the implementation in Apple devices.

- **Wikititude**: Wikitude is a fully functional augmented reality SDK. It has many important features including object recognition and tracking, instant tracking, and video overlay. Wikitude is also a cross-platform SDK, which runs on both iOS and Android.

- **Vuforia**: Vuforia is one of the most popular AR SDKs nowadays. It uses computer vision technique to recognize objects and track images or 3D objects. It is always the first choice for developers to develop mobile and helmet applications. Vuforia provides interfaces of many programming languages and supports native development on Android and iOS. It has integrated with Unity, which makes developing AR games a simple and leisurely work. Figure 2.3 shows the architecture of Vuforia framework.

### 2.2.3 Display Part

Display part is the window for end-users to see final output of the AR system. Three main display methods are now used in current AR applications.
**Optical see-through display**: This technique uses an interface, which is both transparent to see-through the real world and reflective to display the rendered objects. It provides the most immersive experience compared to the other two techniques. Examples including HoloLens and Google Glass that was mentioned before.

**Video see-through display** This technique currently has two solutions. One is using Head-Mounted Display (HMD) devices, which has been popular in the past few years for early AR systems. The other is through the screen of mobile devices, which uses the integrated camera to capture the live world video and combine with the virtual object then display on the screen together. It is now the prominent solution for AR applications on mobile devices.

**Surface projection display** This technique requires an additional projection equipment to render 2D or 3D objects on surfaces from real world such as wall, broad or paper. It’s often impracticable to imple-
2.3 AR Registration

Registration and tracking are the core techniques to make AR technology available. Registration means the procedure of initializing the virtual world coordinate to align with the real-world coordinate. Tracking means correctly evaluating the real-time movement and posture to align the virtual object. Registration is the most important part to achieve augmented reality. The success of registration influences the following AR operations. Bad registration results would affect the functionality of the application. Figure 2.4 and figure 2.5 give two different registration results. Figure 2.4 shows a successful result of registration. We can see the rectangle boards are perfectly adhere with the surface of the notebook. Figure 2.5, however, is an example of failed registration. Those blocks are generated in wrong angle with the surface and thus the pose will still in wrong direction in tracking part and make no sense. Currently, two types of registration methods are commonly adopted, which are sensor-based method and vision-based method. [7]
2.3.1 Sensor-based Methods

Sensor-based registration calculates pose information using several different kinds of sensors-based on various data. [20] present a method, which combines the inertial accelerometers and gyroscopes to estimate the pose and position of the mobile platform. Other approaches like [3] propose a method using magnetometer to estimate the orientation of the device. Another popular registration method is using geolocation, [28] develop an AR game using GPS registration. The game gets the user position data through GPS sensor and generate the virtual 3D object-based on real world latitude and longitude. Besides geolocation sensors, ultrasonic sensors are used as another approach for tracking and registration in [18], the authors use Low-Frequency Ultrasound (LFU) as the audio marker for registration. Table 2.1 shows the comparison of various sensor based methods from [35].

<table>
<thead>
<tr>
<th>Tracking technology</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mechanical</td>
<td>Exactness, low time delay, no vision or magnetic field disturbance, suitable for exact track for small objects</td>
<td>Limited use range</td>
</tr>
<tr>
<td>Magnetic sensing</td>
<td>Low price, exactness, no vision occlusion, good noise immunity, suitable for large field track</td>
<td>Easily disturbed by magnetic field and metal in the environment</td>
</tr>
<tr>
<td>GPS</td>
<td>Suitable for outdoor large field track</td>
<td>Uncertain measuring precision, time delays</td>
</tr>
<tr>
<td>Ultrasonic</td>
<td>Low price, no magnetic field disturbance, light equipment</td>
<td>Easily disturbed by ultrasonic in the environment, low precision in large range</td>
</tr>
<tr>
<td>Inertia</td>
<td>No distance limitation, high speed, no vision or magnetic field disturbance, small size, low price</td>
<td>3 degrees of freedom, drift, not very exactly at low speed</td>
</tr>
</tbody>
</table>
2.3.2 Vision-based Methods

Vision-based registration focuses on estimating posture data from the corresponding feature points in the real world. As it is discussed in section 2.1, there are two ways to achieve registration, which are marker-based methods and markerless AR.

The main procedure of a marker-based registration can be divided into two parts: marker detection and marker pose estimation. I am going to simply introduce them in the following part. In marker detection part, one usually choose easily detectable pattern pictures as markers. The detection method basically contains the following steps:

1. Get the image input.
2. Convert the raw image data to intensity image data.
3. Contour and corner detection using image processing technology.
4. Match the template marker.

After finding the marker, the next step is estimating the pose of the marker. The position and orientation is collectively called pose. As it mentioned before, registration is a procedure, which initializes virtual world coordinates to align with the real-world coordinates. Figure 2.6 shows the basic tasks of pose estimation. The nature of it is finding the transformation matrix between world coordinates and image coordinates. The following mathematical equations and methods of pose estimation are mostly from book [32].

In the marker detection part, one has to calculate camera transformation matrix at first. The equation is:

\[
x - TX,
\]

(2.1)

where \( X \) is a point in world coordinates, \( x \) is its projection in ideal image coordinates. The camera transformation matrix is \( T \) here. Matrix \( T \), contains a 3 \( \times \) 3 rotation matrix \( R \) and a transform vector \( t \). So equation 2.1 can be written as:

\[
x = [R|t] X,
\]

(2.2)

which in homogeneous coordinates is:

\[
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix} =
\begin{bmatrix}
r_1 & r_2 & r_3 & t_x \\
r_4 & r_5 & r_6 & t_y \\
r_7 & r_8 & r_9 & t_z
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
\]

(2.3)
Figure 2.6: The transform between world coordinates to ideal camera coordinates and real camera coordinates. [32]

All the equations and theories are working on an assumption, which is that the camera we use is a pinhole camera. However, digital cameras are the real cameras we use in daily life. Digital cameras map incoming light rays to image pixels. The process of determining the internal parameters, which define the mapping, is called camera calibration. In digital cameras, $p_x, p_y$ represents the offset from the center of the camera coordinates(image coordinates) to the optical axis of the lens. Let $K$ be the camera calibration matrix, the mapping between image coordinates and ideal camera coordinates is written as $x_{pix} = K x_c$.

Written out the equation:

$$
\begin{bmatrix}
  x_{pix} \\
  y_{pix} \\
  1
\end{bmatrix} =
\begin{bmatrix}
  f_x & s & p_x \\
  0 & f_y & p_y \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x_c \\
  y_c \\
  z_c
\end{bmatrix},
$$

(2.4)

where $s$ represents the rotation of pixel square, $f_x$ and $f_y$ are the width and height of the pixel.

Now, the extrinsic matrix $T$ and intrinsic matrix $K$ are all carried out. In marker detection, we got four corner points in image coordinates $x_1, ..., x_4$. For each corner point $x_i, i=1,2,3,4$, the following equation is always true.

$$
x_i = K T x_i
$$

(2.5)
CHAPTER 2. BACKGROUND

If we expand equation 2.5, it is expressed as

$$
\begin{bmatrix}
    x_i \\
    y_i \\
    1
\end{bmatrix} = 
\begin{bmatrix}
    f & 0 & p_x & 0 \\
    0 & f & p_y & 0 \\
    0 & 0 & 1 & 0
\end{bmatrix} 
\begin{bmatrix}
    r_1 & r_2 & r_3 & t_x \\
    r_4 & r_5 & r_6 & t_y \\
    r_7 & r_8 & r_9 & t_z \\
    0 & 0 & 0 & 1
\end{bmatrix} 
\begin{bmatrix}
    X_i \\
    Y_i \\
    Z_i \\
    1
\end{bmatrix}
$$

(2.6)

If we multiply $KT = M$

$$
\begin{bmatrix}
    x_i \\
    y_i \\
    1
\end{bmatrix} = 
\begin{bmatrix}
    m_1 & m_2 & m_3 & m_4 \\
    m_5 & m_6 & m_7 & m_8 \\
    m_9 & m_{10} & m_{11} & m_{12}
\end{bmatrix} 
\begin{bmatrix}
    X_i \\
    Y_i \\
    Z_i \\
    1
\end{bmatrix}
$$

(2.7)

We can get the location of these four corners in world coordinates. Because the system has eight equations and six parameters to solve, the transformation matrix can be calculated. Therefore, the pose of the marker can be determined, and thus the registration is successfully done.

[14] propose a marker-based registration system. The prototype uses QR code as the registration marker for the AR system. Its camera would search for the QR code in every frame. After locating the QR code successfully, the virtual content would be displayed at the same position of the QR code. However, a binary QR code would influence the immersive effect for AR system, since a QR code is obtrusiveness in our real-world environment. [22] introduce a method which uses infrared marker and infrared cameras to do registration.

Although this method solves the problem of visibility of markers, it requires an external infrared camera to detect the marker, which is unpractical for universal implementation. Research on markerless AR is becoming popular recently, due to the development of computer vision algorithms. [33] propose another vision-based registration method using nature feature points from arbitrary planes, which does not require neither sensors nor markers. It is an example of markerless AR. Its basic idea is to track the KLT-feature of every plane detected. Then, calculating projection matrices of each plane. Next, the geometrical relationship between the planes and the camera can be estimated by those projection matrices. Finally, the virtual component can be generated on the camera, the registration is complete. However, the stability and registration accuracy is quite low. [13] introduce a method based on Visual Simultaneous Localization and Mapping (VSLAM), which reconstructs the 3D space around the camera and then produces a good registration accuracy.
2.4 AR Applications

AR has vast applications that can fit on many realistic scenarios. Most of them are implemented on mobile smart devices. Because through mobile devices, the product can reach millions of users. In the meantime, the number of AR supportive devices is growing. In 2017, Google and Apple both rolled out new AR SDK, ARCore and ARKit, which enable developers to make thousands of AR applications with high level performance and to embrace two biggest mobile operating systems with millions of active users. They are revolutionary progress in AR technology and favor us more than any time. In this section, several popular applications of AR will be discussed.

2.4.1 Tourism and Navigation

The ability to display information with real world makes AR very useful in tourism and navigation. Figure 2.7 shows an application of AR on a driving navigation system. Fritz et al. [12] developed a video see-through AR system, which enables the tower viewer to display multimedia interactive information to make the tour experience more attractive. Oliveira et al. [10] designed an indoor navigation system using AR. They implemented markers at the interesting location points and generate arrows on the marker. Other than these researches and applications, AR has abounding applications in tourism field, including enhancing booking experience, re-living historic architectures, building interactive museum tour, etc.

Figure 2.7: An AR navigation application. The orange arrows indicates the direction of the road [23]
2.4.2 Training and Education

Training and education are always the leading edges of AR, since AR can mix virtual environment with real environment. Reality is very important for training, but sometimes reality means danger. For example, a fighter jet pilot should be skillful in flying planes and other tough military situations, it is necessary to have drills often. Enhanced reality, which simulates the drill with virtual component, would help them become familiar with every circumstance and keep them safe at the same time. Back to the early beginning, AR was born to be as the technology for training pilots of U.S. Air Force. The first AR systems, Virtual Fixtures, developed by U.S. Air Force’s Armstrong Laboratory in 1992. [29]

AR is interactive and digitally manipulable. These features enables it to be a great tool for education. In [15], Gupta et al. listed several existing and possible AR education applications. In their overview, AR was used for astronomy learning through showing interactive celestial bodies, which make students understand celestial bodies better. Another example, AR was used to design a physics learning application, in order to let students have a better understanding of electromagnetism concepts. There are also possible applications in education. An AR displayed 3D DNA model can be a great usage for understanding its sophisticated structure in biology field. Furthermore, AR could be used for simulating some dangerous chemical experiments.

2.4.3 Assembly and Maintenance

The ability to bring real environment through video see-through display of AR can help assembly and maintenance run in an effective and protected way. Brizzi et al. [4] combined telesoperated industrial assembly system with AR. The system uses a wearable devices for operators to control a remote robotic mechanical arm. The arm executes assembly maintenance operations. The real video stream is transmitted to the head-mounted device to the operator. AR generates information on the head-mounted display, to imply the operator the result and performance of the operation.

Cordonnier et al. [9] explore possibilities to use augmented reality in network equipment maintenance. They researched and experimented the AR application in three aspects. First, in the network fault diagnosis, a remote expertise interaction application is developed. The maintenance expertise could discuss the problem of the network with local stuffs. AR is used to indicate on the image to show points worth noticing on. Second, after the algorithm recognized the network panel, the AR information of the data
would be demonstrated, which supports the maintenance very good. Last, AR is used to display the underground cables so that the technician could locate the fault location easily and accurately. The output of augmented reality cables is shown in Figure 2.8.

2.4.4 Advertisement and Game

The number of smart phone users in the world will be 2.5 billion in 2018. The huge group of users are potential commercial delivery targets. In 2006, [8] brought interactive product commercials on mobile phones, which was a leading utilization at that time. Many companies applied AR to product advertisements nowadays. Figure 2.9 is a AR commercial demo of an ice cream product.
CHAPTER 2. BACKGROUND

AR also plays an important role in the game industry currently. It allows game players play video games in the real-world environment. The immersion with game and real environment makes the game more attractive and interactive since players can explore in the real world. Figure 2.10 shows a example game made marker-based AR. The virtual 3D map is generated using Vuforia object detection, it found out the poker and generated the 3D object on it. Beyond marker-based AR game, Niantic released a geolocation-based treasure-hunting AR game Pokémon Go in 2016, whose daily active users reached millions after one month. [27] It is a great example that AR has the power to play an inseparable role in the video game industry.

Figure 2.10: A marker based AR game made by Vuforia

However, games made with AR still have some issues to be solved. For example, most of the Geo-location-based AR games right now are using GPS to locate the player. Which would have a deviation in some weak signal areas such as indoor buildings, basements, and remote mountainous areas. And they are not suitable for all kinds of weather. Hunting a Pokémon at outside environment in winter days of a Nordic country may not be appealing anymore. There is a huge need for indoor Geo-location-based game, but not too many people are working on it right now.

Developing an indoor Geo-location-based game is similar with outdoor Geo-location-based game. The main difference is that the two are using different localization methods. Unlike being localized by GPS satellites, indoor localization has variable methods. The next section would focus on indoor localization.
2.5 Indoor Localization

Indoor localization has been a hot topic in recent years. It came out to solve the problem that GPS signal would be shielded by building infrastructures or be interfered by noise. It gives the user an accurate positioning service inside buildings. Indoor localization, like augmented reality, also benefits from the wide availability of mobile devices, which comes closer to us.

2.5.1 Localization Methods

To get an accurate localization result, several methods have been researched in the past few years. Jiang D. [11] listed five main techniques in indoor localization field including labeling, triangulation, dead reckoning, fingerprinting and image. The following content elaborates these methods in detail.

2.5.2 Labeling

This method requires pre-installing labels in indoor environment. Many entities can be used as labels, such as QR codes [6] and Radio-Frequency Identification (RFID) [2]. The user gets connected with the label through camera or RFID receiver. Then, the localization is based on finding the closest label, this can be achieved by recognizing the QR code or analyzing the signal strength depends on different method. In consequence, the accuracy of labeling indoor localization heavily depends on the density of the labels, more labels give higher accurate localization service.

2.5.3 Triangulation

This method is similar with labeling method. Bluetooth [31] devices and Wi-Fi devices [19] are the mostly used to set as reference points. The main difference between triangulation and labeling is that triangulation calculates the location based on data from three reference points. By analyzing the data, the location and direction relation of user and reference points can be estimated, since the reference points are pre-installed and their position is fixed and acknowledged. In consequence, the location of the user can be determined. Compared with labeling method, triangulation requires fewer pre-installed reference points, which can save plentiful cost.
2.5.4 Dead reckoning

This method is a localization method, which determines one’s current position based on its previous position and the speed and time of the motion between. Pedestrian Dead-Reckoning (PDR) [37] is currently a prominent method. Dead reckoning requires other indoor positioning method to get the initializing location. After getting the start location, it would apply data from the embedded gyroscope sensor to calculate the motion data and then calculate the current position of the user. The precision depends on the accuracy of the inertial sensor applied.

2.5.5 Fingerprinting

This method traditionally relies on Received Signal Strength Indication (RSSI) [36]. Basically, it collects the location data and the correlated Wi-Fi fingerprint and store them in the database. When the system gets a localization request, it sends the current fingerprint to the system. Then, the system finds the most similar fingerprint in the database. Next, the system send back the location, which is correlated with the fingerprint. Since Wi-Fi devices are common equipment in an indoor environment, fingerprinting method does not need pre-install any external devices.

2.5.6 Image-based

This method is a computer vision-based method. The method requires getting a bunch of features beforehand. When a user takes a picture of that space, the method would find the matching between the feature points in taken picture and those in cloud points database, and calculates the camera pose at the same time. Image-based localization is a trendy topic in these years. For example, [25] proposed a indoor localization method using 3D Structure from Motion (SfM). A 3D model was built based on 3D SfM. Then, the feature points and approximate nearest neighbors are implemented to expedite the discovery of feature coherence.

To build the point cloud, SfM technique is used. SfM is a computer vision term, which means the technique of estimating the 3D structures from 2D images. [26] introduces basic steps for building a 3D model out of 2D images sequence, which are features extraction, features matching, camera calibration, and position recovering. First, one popular algorithm for features extraction nowadays is called Scale-Invariant Feature Transform (SIFT). One of its main advantages is that it is scale invariant. This advantage allows to detect image features in different scales, rotation angles, and translations.
Then, one finds the same feature point in different images using features matching technique. In [1], Snavely et al. find the matching points using SIFT feature matching and Random sample consensus (RANSAC) as verification of similarity. Next, camera calibration is used to determine the camera intrinsic parameters. The detail about it is addressed in section 2.3.2. Finally, the position of points are computed using triangulation technology and the point cloud is built.
Chapter 3

System Design and Environment

This chapter talks about the whole system overview for building an indoor AR game. Section 3.1 presents the concepts and considerations during designing the system, as well as an example game scenario. Section 3.2 describes the basic architecture of the system, including both client architecture and server architecture. Section 3.3 introduces the software development tool used in this thesis, and 3.4 addresses the hardware environment in the implementation part.

3.1 System Design

Section 3.1.1 talks about the reason why image-based localization method is chosen as the localization method of this thesis. Section 3.1.2 lists several fundamental requirements of the system. Section 3.1.3 presents an example scenario about the whole procedure of the game.

3.1.1 Localization Method

When choosing the localization method of an AR game, several cases should be considered. First, the indoor environment should contain few or none artifacts. For example, an AR treasure hunting game using QR code labeling methods would depreciate the realism of the game. Since a semantically correct world would not have any QR code inside of it. Second, the game should be available for both Wi-Fi connection and cellular connection. It is unreasonable to force the user to change data connection mode every time they open the game. Under the circumstance when the device may not have the connection with Wi-Fi devices, methods like triangulation and fingerprinting are not suitable for the game development any longer. As consequence,
image-based localization is an ideal method for an AR treasure hunting game. Because image-based method does not require installing any artifacts in the environment. In the meantime, it is suitable for any data connection mode as it does not specifically use Wi-Fi data, and the AR application always uses the camera to do registration and tracking, which could also be used as the input of the localization system.

3.1.2 System Requirement

A normal indoor AR treasure hunting game should be played inside buildings with more than one player. It should be a mobile game that runs on cellphones and tablets. To reach the essential features of this game, the following requirements should be taken into consideration:

1. Localize the player in the indoor environment: one of the core features of the game is locating the player accurately in the indoor space. In this thesis, an indoor positioning service offered by VimAI Oy is used for indoor localization. VimAI provides highly accurate indoor localization and navigation services by mainly using image-based method. By applying their algorithm to the image, VimAI has the capability to calculate where the user locates accurately.

2. Display treasure using AR technology: some game component should be rendered by AR, so an AR SDK is mandatory in the development. In the game development procedure, Vuforia is chosen to build up the AR features in the game.

3. Deploy a game data server: to play the basic treasure hunting game, a server should be implemented to transact data over each client. To be more specific the server take REST request to store or retrieve position data from the database, and send response to the client.

4. Connect every player for multiplayer mode: multiplayer mode should be a primitive feature of the game, if one player have found the treasure, other player would lose the game. A method for synchronizing states of players should be implemented.

5. Reach the widest users: the game should be able to launch at most of the mobile devices currently.
3.1.3 Example Game Scenario

Three players are on the 2nd floor of Open Innovation House\(^1\), which is an indoor space with VimAI indoor positioning service support. Player1 opens the game on his phone. Then, Player1 walks to the coffee machine and stops there, he presses the “HIDE” button in the game. The game sends the current image binary data to VimAI server, then the server sends location back. After getting this location data of Player1, the game immediately sends this data to the game data server, the server stores the position data in the database.

Now, Player1 could tell others that he has hidden the treasure successfully, the other two start to search the treasure in the space. They open the game on their mobile phones, and press the “HUNT” button, the game starts to send the image data to VimAI’s server consistently, and the game tells them if the treasure is close or far through the rotation speed of a “radar” on the screen. If someone gets really close to the treasure, the radar would rotate rapidly, and vice versa. Player2 goes towards to the coffee machine, and she gets the radar rotation faster and faster. On the contrary, Player3 goes the opposite way, so his radar rotation speed becomes slower, he gets further to the treasure.

Finally, Player2 walks close enough to the treasure, the game gets her position and finds out it almost matches the treasure position, the treasure would appear on her phone screen. Then, all she has to do is to fire a “CATCH BALL” to get the treasure by a simple swiping. Player2 swipes the screen and the ball hits the treasure. She wins the game, and at the same time, Player3 loses the game, the game is over.

3.2 System Architecture

After analyzing the system requirements. The client-server model is chosen as the basic system architecture. The whole system is divided into two parts, a mobile client for playing the game activity, including getting image frames, locating the players, rendering AR treasures, and transmitting data, and a backend server for managing position data. Figure 3.1 shows the system architecture in network perspective. In the model, server of VimAI is also included to clarify the data transmission procedure. The client and server communicate with each other by RESTful web service. The client uses GET method to retrieve the data from the server, and uses POST data to send the new data to the server.

\(^1\)Address: Maarintie 6, 02150 Espoo
3.2.1 Client Architecture

The client is supposed to be an AR game that runs on multi-platform mobile devices developed using Unity and Vuforia. The architecture of the game client is shown at Figure 3.2.

The game controller is the main control center, which manages game data-flow, game logic control, game scene loading, game UI control and audio control. The camera controller manages every camera activity. It converts raw camera input to server-supportive file types. To reduce the client server communicating time, the camera controller compresses the image data first. The image data handler is the handler to make communication with the localization server. The data handler uses Unity WWW class to send POST request to VimAI’s server. Then, the server runs the algorithm to calculate the position of the image and sends back the position data back to the client. Next, the client’s status handler examines the server input data-flow. If the status is Hide, which means someone is trying to locate itself and hiding the treasure. The handler sends the position data to the game data server. On another situation, if the status is Hunt, the handler transfers the position data to the distance handler to do the next operation. The game examines the player find the treasure, if and only if the distance between the first player’s
position is less than a threshold $D$. The ranges of $D$ will be discussed in the next chapter. As long as the distance is smaller than $D$, the game controller calls the AR generator to render the AR treasure on the screen, and it will also play an audio to note the player that he has got the treasure. The last thing it has to do is hitting the treasure by missile. If the missile hits the target, the player wins the game. The AR generator and missile controller is controlled by network controller. The network controller gets every player’s game status, once a player got the treasure, every other player lose the game and cannot play any more.

### 3.2.2 Server Architecture

To design a RESTful database server, a Spring Boot + MongoDB framework was implemented. There are several reasons for choosing this framework. First, since only a light-weighted RESTful web service is needed in the whole system, it is recommended to use an easy-implement architecture to build the web application. Spring Boot is currently a practical and popular web service framework. Second, the data transmitted in the system is position data. It contains an unique ID to identify each data, and X,Y,Z data of the 3D coordinate of each position. It is unnecessary to use a relational database such as MySQL to handle this substantive data, and thus I chose the simple and flexible database called MongoDB as the backend database.

Figure 3.3 shows the architecture of the position data server. The client side communicates with the server by sending HTTP REST request. In
the web application block, a REST controller handles all REST requests. For example, when the client sends a GET request, the controller maps the request with the method through url, then retrieve the latest position data from the database and send back the data in JSON format. Another example, when the server gets a POST request by the client. The controller manages the input POST parameters, maps them to a Java class and store the data in the database. The Java class block is used for resource representation. The position data has to be modeled as a Java class to do the database mapping. The last block is a class to notify Spring to make connection with the database. These three blocks work together to handle all requests from the client, write and retrieve data from the database and send response back to the server.

### 3.3 Software Environment

There are several excellent game engines for game development. Unity and Unreal are two most popular developing tools over them. Both of them are cross-platform, AR supportive game development tools. In the thesis, Unity is chosen as the AR game developing tool, because Vuforia was used as the AR SDK and Unity has a better integration with Vuforia than Unreal. Vuforia enables Unity developers to create attractive AR experiences and reach the most users. It also allows developer to deploy AR project on a
variety of handheld devices and headsets, such as iOS and Android. Besides
the advantage with AR development, Unity has several additional benefits.
First, Unity has a massive and supportive developer community, which can
help the development very smoothly. Because you can ask for advises and
solutions from the community if you have obstacle during the development
procedure. Next, Unity has a good ability to develop multiplayer games.
Unity provides built-in network managers for multiplayer games. The thesis
implementation is benefited from this function, since it is needless to worry
about the low level network issues in the multiplayer network. Last, Unity has
an abundant assets store, which provides all kinds of resources for building
a game. The treasure hunting game requires some resources like 3D model
with animation, soundtrack, and UI fonts. Those are easy to access from
the Unity assets store. In conclusion, Unity is a mature and suitable tool for
AR development. Development using Unity and Vuforia will accelerate the
implementation and keep the quality of the prototype.

3.3.1 Unity Components

In the coming implementation, Unity is the main development tool. It is
necessary to introduce several essential components in order to understand
the developing procedure. This section proposes five important components
in Unity game development.

1 **Scenes**: The game environment and menus are displayed in Scenes. A
single scene is built with game components, including Prefabs, audios,
UI interfaces, and programming scripts, etc. A scene could be a level
of a game. It could also be the main menu of the game. When the
scene starts, it loads all the resources of that scene into the memory
and generates the game environment defined by scene.

2 **GameObjects**: The basic unit of a game scene is GameObject. In
Unity, GameObject is the fundamental element in development. The
concept of GameObject is broad. Everything that appears in the game,
i.e., cameras, sounds, 3D objects, lights, or effects, are GameObjects.
Unity gives a convenient method to reuse predefined GameObjects.
Unity has Prefab type, which allows the developer store the complete
GameObject.

3 **Components**: Components provide properties of GameObjects. In or-
der to describe a GameObject, it has to be assigned with Components.
Components are the information of a GameObject. Several common
components are transform components, rigidbody, and scripts. With a
specific example, a 3D ball has a rigidbody component, which gives it a physical mass. When the scene starts, the ball falls on the ground since the rigidbody gives it gravity.

4 **Scripts**: Unity uses Scripts to control the GameObject. Scripts is considered as behavior components in Unity. Two programming languages are natively supported in Unity, which are C# and UnityScript. Scripts have to be attached to GameObjects, so they can be executed on them. Variables in scripts control the data of Components. Scripts use methods to control the behavior of Components and GameObjects. The main task in this thesis is writing scripts to make every part work properly.

5 **Plugins**: Plugins are pre-written codes outside Unity, which allows Unity to extend the ability to finish specific tasks. It can be easily installed in Unity. The Vuforia plugin enables developer to access Vuforia features in Unity directly, which is used in the implementation part of the thesis.

### 3.4 Hardware Environment

One of the goals of this thesis is to implement the game at most of the mobile devices. This requires me to implement the game on the platform with most users and mediocre mobile phones. Two models of Android devices are chosen as the hardware environment to implement the game. Table 3.1 gives the review of the specification of the two phones.

<table>
<thead>
<tr>
<th>Phones</th>
<th>LG Nexus 5</th>
<th>Huawei Honor 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Announced Year</td>
<td>2013</td>
<td>2016</td>
</tr>
<tr>
<td>RAM</td>
<td>2GB</td>
<td>4GB</td>
</tr>
<tr>
<td>Internal Storage</td>
<td>16GB</td>
<td>32GB</td>
</tr>
<tr>
<td>Camera</td>
<td>8MP/autofocus</td>
<td>12MP/laser autofocus</td>
</tr>
<tr>
<td>OS</td>
<td>Android 6.0</td>
<td>Android 7.0</td>
</tr>
<tr>
<td>CPU</td>
<td>Quad-core 2.3 GHz Krait 400</td>
<td>4x2.3 GHz Cortex-A72 &amp; 4x1.8 GHz Cortex A53</td>
</tr>
<tr>
<td>GPU</td>
<td>Adreno 330</td>
<td>Mali-T880 MP4</td>
</tr>
</tbody>
</table>
Chapter 4

Implementation

This chapter tells about the implementation of the game. Section 4.1 tells about capturing the input image data. Further modification of the data is done to improve the game performance. In section 4.2, details of building the RESTful server are addressed. Then, the process to manage the game scenes and other objects are touched over in section 4.3. Section 4.4 talks about the method to calculate the distance between the player and the treasure, as well as how I visualize the distance. Then, in section 4.5, the implementation of camera coordinate system is brought in. Section 4.6 proposes the fundamental of Unity multiplayer networking and its application in this game. In the end, section 4.7 discusses how to create and capture the treasure.

4.1 Image Data Modification

In the development part, I implemented the image data handler first, since taking picture is the most important part of the game. The logic of this part is capturing the 2D pixels data from the live video stream, processing the data to meet the basic requirement for other system components, and finally passing the data to VimAI indoor localization server.

Unity offers WebCamTexture class, which is a 2D texture object rendered from live video input. However, Vuforia does not support rendering AR objects on it. Fortunately, Vuforia offers a Vuforia.Image class to expose camera input frames. VimAI indoor localization server supports RESTful web service. One main operation of it is to receive POST request. The request requires two mandatory parameters, which are image binary data and building ID. VimAI scans the indoor environment first to build a 3D point cloud model in advance, so every building has an unique ID to distinct different point cloud models. After getting the building ID and image data,
a **POST** request can be generated.

### 4.1.1 Image Compression

During the implementation procedure, the data transmission efficiency caught my attention. It turns out that the uploading time of a single image is too long. The average size of captured image binary file is around 400KB, which is oversize for data transportation. The game is designed to send the image byte data to the server once in 3 seconds, but most of the time the transferring time of image data is more than 3 seconds. It caused a serious problem, which is the game cannot locate the player even though the server has a perfect performance and localization function.

Compressing the image data is the first method I came out with. In section 2.5.1, it was mentioned that the image-based localization has a feature called scale-invariant. It indicates that the output in a range of compression would not discourage the finding of feature points. The question is that how much the image data could be compressed so that the localization will be as successful as before and the sending speed will be faster at the same time.

A benchmark was created to analysis the relation between image compression rate and the success rate of localization. Several sets of images were taken in the experiment. Different locations were chosen to take pictures in different compression rate. The example image sets are shown at Figure 4.1.

![Image data in different compression rate](image)

**Figure 4.1:** Image data in different compression rate

In Figure 4.1, each column represents a static location example in Open Innovation House, and each row represents a range of image quality. From top to bottom, the image quality is 1, 25, 50, 75, and 100. 1 is the lowest quality one could make, which means the image is compressed in the worst quality. 100 means the image keeps the original data size. In the experiment,
250 images from 50 locations were captured. These image sets were sent to the localization server and the localization result from the server is shown at Table 4.1.

Table 4.1: Localization result table

<table>
<thead>
<tr>
<th>Image quality</th>
<th>1</th>
<th>25</th>
<th>50</th>
<th>75</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localization rate</td>
<td>0</td>
<td>74%</td>
<td>84%</td>
<td>90%</td>
<td>90%</td>
</tr>
</tbody>
</table>

In the result table, the localization server cannot locate the image compressed by image quality at factor 1. The quality of those images is too low to extract feature points. 5 images got all the result of “location failed” in every compression rate. It is normal because the server can fail to localize the position due to lack of feature points matching or other reasons. In the total 50 example locations, 37 images are successfully located at the image quality of 25. 42 images are successfully located at the image quality of 50, and 45 images are successfully located both in factor 75 and factor 100. Figure 4.2 shows the average image size in different image qualities. Image compressed by factor 75 is already much smaller than the original size, the compression ratio is 4.97. It is an impressive outcome for compressing images. In the end, the image quality is set to 75 so that the localizing result cannot be influenced, also the network behavior rises up.

Figure 4.2: Image size in different compression rate
4.2 Position Data Server

After acquiring the position data, the game goes to the next part. The game status handler checks the current status of the game. In this game, Hunt and Hide are two statuses the player can choose. The handler sends the data to the data storage server if the player is playing the Hide mode. In the other scenario, the status handler passes the current position to the distance handler, which calculates the distance between the treasure position and the current position of the player. These crucial scenarios require a data server to store and retrieve the game data, while every device can transfer data with it.

In order to build a data server that fulfills the requirement, a RESTful web service seems to be the most applicable choice. First, the game client and the server only transfer data between each other. The server only manages game data and the client only runs game functions. The client and the server are self-reliant. Then, the server should be stateless, which means each request from the client should be independent and isolated. Finally, a RESTful web service is easily extensible. If the game expects more features from the server, it is simple to extend the api and functions in the server side.

Spring Boot is used as the basic framework in the game server implementation part. Spring framework has several advantages that makes it the most appropriate tool for developing RESTful web server. First, Spring has sufficient support for RESTful web services. It offers dedicated java annotations for RESTful development, including @RestController and @ResponseStatus. The @RestController handles HTTP requests and map those requests to dedicated Java classes or methods. Next, Spring supports parameterized URLs by using @PathVariable to extract data from the URL. At last, Spring Boot is easy to connect to MongoDB and is simple to deploy to cloud platforms. A position data server is built using the architecture mentioned in 3.2.2.

The server accepts two query URLs, a /getPosition URL receives a GET request sent by clients and returns a JSON data contains position id and x,y,z values, and a /postPosition URL gets a POST request and maps the position data from the request body to the database. The server was deployed on Heroku\(^1\) to enable the game client transfer data via WWW.

\(^1\)Heroku is a cloud application deploying platform. url: https://www.heroku.com/
4.3 Scene Control

The last section told the game contains more than one scene, and in section 3.3.1, the importance of scene in Unity was talked about. A single scene contains all the building resources for a game environment. However, a game may have more than one scene. A scene manager is then needed to handle the switch between each scene. In this thesis, the game has three scenes, which are Menu scene, Hide scene, and Hunt scene. Menu scene is the initial interface after player opens the game. It contains the entrances to other two scenes, an exit button to shut down the game, and a setting menu. In the setting menu, the user can manipulate the volume of the game, and choose the building id depending on his location. Hide scene is for the player hiding the treasure. It contains a button to take picture and send position data to the server. Hunt scene is the place, where most of the players looking for the treasure, capturing it and competing with each other.

4.4 Calculating the Distance

The distance handler is responsible for calculating the distance between the current position of the player and the position of the treasure. In section 3.2.1, a threshold $D$ was created to examine whether the player catches the treasure. This section mainly answers the question that how the distance is measured and how the threshold $D$ is set.

4.4.1 Euclidean Distance

Euclidean distance is chosen as the measurement distance between two points. In mathematics, Euclidean Distance of two points represents the straight line distance between the two points.

[34] defines Euclidean distance as following: in Cartesian coordinates, if $p = p_1, p_2, ..., p_n$ and $q = q_1, q_2, ..., q_n$ are two points in Euclidean n-space, then the distance $(d)$ from $p$ to $q$, or from $q$ to $p$ is given by the Pythagorean Formula 4.1

$$d(p, q) = d(q, p) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

\[= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \quad (4.1)\]
CHAPTER 4. IMPLEMENTATION

The response data from VimAI localization service is organized in JSON format, which can be found from Appendix A. It contains position data in three dimensional space, direction data in three dimensional space, rotation data in quaternion space, an integer represents the areaID, a timestamp field to identify each response, and an attribute to represent the statistic data is in the end.

The position of the treasure is defined in two dimensions, which is the X and Z value in the position attribute. Let P represent the location of the player, and T represent the location of the treasure. The distance between P and T can be calculated by Formula 4.2.

\[ d(P, T) = \sqrt{(P_x - T_x)^2 + (P_z - T_z)^2} \]  

(4.2)

4.4.2 Threshold D

The location of the treasure is a point in a two dimensional space. It is impossible to get the exactly same point when the player locate itself. A threshold D is created to confine a distance around the treasure hiding point. D cannot be too large, which will miss the original hiding place dramatically. D cannot be too small either, the player will hunt the treasure very hard in that case.

![Walking Speed](image)

Figure 4.3

Figure 4.3 displays a special case may happen in the game. F1 represents one request of localization, at that time, the distance between the player and the treasure is a bit longer than D. Then the player walks in the direction toward the treasure. There is a time between two requests of localization, during which the game does not take picture and send localizing request. Next, the player arrives F2, where the distance to the treasure is just further than D. At this position, the next localizing request has just been sent. In this extreme case, the player cannot find the treasure even though he has passed the treasure. The threshold D has to be set in a reasonable value to prevent player from missing the treasure. Formula 4.3 shows the relation of
walking distance and the threshold distance.

\[ \Delta t \cdot V_w < 2D \] (4.3)

In the Formula, \( \Delta t \) represents the time interval between two sequential requests, \( V_w \) is the walking speed of the player. The average walking speed of human adults is between 1.3-1.5 meters per second [5]. In the game implementation, \( \Delta t \) is 2 seconds, so according to the formula, \( D \) is set as 1.5 meter naturally.

4.4.3 Visualization

Developing a treasure hunting game is a hard job. One of the most challenging tasks is designing the game strategy. Imagine the player gets its position successfully, how does the game tell the player where to move? In this section, several methods to visualize the treasure position will be discussed.

The response data contains the direction of the player. The first approach is to visualize the relative direction between the treasure and the player. To make that possible, an arrow, which targets to the treasure position, is generated on the screen. Whenever the game gets the current position of the player, the arrow would target to the treasure hiding position. It appears that the method is unrealistic during the experiment period, because the player can find the treasure easily and rapidly. The arrow informs too much information about the final position of the treasure. All the player has to do is to locate once and follow the arrow then he gets the treasure. There is no fun if this method is used in the game.

After the failure in the first method, all approaches about visualizing the direction to the treasure were rejected. Another method is to visualize the distance between the treasure and the player. By acquiring the information of the relative distance, the player has many options to move and can find the right way to approach the treasure.

First, a floating rectangle board was created on the screen. This board informs the player about the distance by changing its color. However, after several experiments, this method was abandoned as well. Because if the player is not told the meaning of the color, he would have no idea about the relation between the color and the distance. For example, the game sets the original color to be black, which is (0,0,0) in RGB color mode, and it increases the R value when the distance decreases. Most of the players had difficulty to realize the red color means the distance between them and the treasure. Another problem is that most of the testers found the change of the red color is too slight to be noticed. So, this method was also discarded.
in the end.

Finally, a method to visualize the distance by changing the speed of a rotating “radar” was implemented. A rotating circle was created on the screen, it rotates like a radar detecting the treasure. The closer the radar is to the treasure, the faster the radar rotates. This design came out to be easy to understand for the testers, who commented it is clear of the relation between the speed of “radar” and the distance. Figure 4.4 shows three kinds of visualization methods discussed in this section.

(a) Arrow  (b) Color  (c) Radar

Figure 4.4: Three kinds of distance visualization

4.5 Vuforia World Center Mode

This section introduces the Vuforia world center mode, which controls the game camera behavior. The camera and AR objects have a relative coordinates, world center mode is used to define how the coordinates are translated to world coordinates in Unity.

In Vuforia, AR objects are defined as Trackables. The inheritance of Trackable class is shown in Figure 4.5. Currently, Vuforia provides four kinds of world center modes, which are SPECIFIC_Target, FIRST_TARGET, CAMERA, and DEVICE_TRACKING. According to [17], each type has different definition of translation from camera coordinate to world coordinate. The first mode, SPECIFIC_Target, the user defines the world coordinate center based on a single Trackable. The user can set the center of the Trackable as
the center of the world coordinate or set any point, which has the fixed relative translation of the Trackable. The second mode is called FIRST\_TARGET, which is easy to understand from the name. In this mode, Vuforia uses the first Trackable that appears in the view as the world coordinate center. The third mode CAMERA does not define a world center, what it does is to move the Trackables with a fixed relations with the AR camera. This mode always sets the AR objects appears in the same place of the camera, which is not suitable for the situation of the thesis. The last mode is DEVICE\_TRACKING. This mode defines the Unity world coordinate center by tracking the pose of the device.

In this thesis, the goal is to implement a markerless AR game. In other words, there is no Trackable in the game. The method talked about in section 3.1.1 is rendering the AR objects based on the current position of the player. Therefore, the most suitable world center mode is DEVICE\_TRACKING mode.

### 4.6 Multiplayer Mode

One of the fundamental requirements of this game is implementing multiplayer function. Unity offers a set of powerful multiplayer and networking tools, which meet the requirements and are easy to implement. Unity provides two different levels of network API for different demands. The High Level API contains necessary features of a primitive game networking, which is used for normal multiplayer games. The Low Level API or the Network-Transport API allows developers to build own network infrastructure for complicated multiplayer games. In this thesis, the game only needs game state communication between the server and the clients and several elemental functions, and thus the High Level API was chosen.

In the High Level API, Network Manager manages the whole network-
CHAPTER 4. IMPLEMENTATION

...ing factors in a multiplayer game, including game state management, spawn management, scene management, matchmaking and, etc. Game state management and spawn management are currently the most important features to implement on the game.

Unity multiplayer networking offers three kinds of networking modes - client mode, server mode, or host mode. Every game must start with a server, so game runs on client mode can be able to connect to the server. Host mode means the instance game runs in server mode and plays as in client mode at the same time.

Converting a game to Multiplayer mode requires player Prefab setup. Each player Prefab must have a Network Identity. Network Identity is the core part in Unity Multiplayer. Each player should have a unique identity in the game, so the server can be aware of the existence of the player object. In this game, the player Prefab contains an AR camera for taking picture and rendering AR objects, and a player info object which stores the game state of the player. When a new player joins the game, the server will be aware of the new player and get its state at the same time.

![Figure 4.6: Unity Network remote action](image)

Figure 4.6 shows the remote action flow of Unity multiplayer networks. These remote actions shown in the diagram are called Remote Procedure Calls (RPC). Multiplayer networking offers two kinds of RPCs, one is called Command, and the other is called ClientRpc. Command is a type of RPCs,
which is called from client and runs on the server. ClientRpc is called from server and runs on all clients.

Unity uses scripts to let developer control game objects, including network behaviors. Game state synchronization is the most important function being used in the game development. One essential task of game state synchronization is to determine, whether a winner exists and inform every player if someone has captured the treasure. To synchronize the state to every player, a method has to be called by the game every frame to inspect if a player wins the game. Fortunately, Unity offers a function called Update(). This function is called in every frame, a frame here represents a screen update. Then a method to inspect the game state is put in the Update() block, so the game checks the state in every frame. If the local player has got the treasure, the Update() function would detect the change, and modify the change to a boolean value. Next, a Command function is called in the local client to inform the server the state has been changed. As it is mentioned before, the Command function is called in client and runs on the server. When the server gets the information, the server will calls a ClientRpc function on every other client that they are not able to capture the treasure in time and the lose the game. Finally, every other client gets the information and execute the relevant function to end the game. The whole procedure of game state synchronization is done.

In the game implementation part, all the multiplayer functions are built by Unity Multiplayer. One crucial function in the game is to synchronize the game state in every playing devices. If someone gets the treasure, other players would not be available to play the game anymore. A class called PlayerController is created to control the player network behavior. In this class, a boolean variable isPlayerWin is set and is given a false default value. The Update() function checks if the treasure is found in every game frame. If the treasure is found by one player, the Update() function in this player's client would detect the change, and executes the Command function called CmdPlayerWin(), as it is mentioned before, CmdPlayerWin() is called in client and runs on the server. Inside CmdPlayerWin(), the ClientRpc function RpcOthersLost is inserted. Once the CmdPlayerWin() is called and runs on the server, the server will call the RpcOthersLost and runs it on every clients. So, every other clients would get the “game lost” information as soon as one player wins the game.
4.7 Treasure

Now, the game has everything prepared for the treasure. The game has the ability to localize the player, store and retrieve the location data of the treasure in the cloud, and connect each player and synchronize their states. The following task is to settle the treasure in this game.

4.7.1 Treasure Placement

The first problem about the treasure during the implementation is that how to set the treasure in the game. To be more specific, should the treasure be put on the ground virtually of the indoor environment or not. Putting the treasure on the floor virtually is workable, because Vuforia has Ground Plane detection, which enables developer to place virtual objects on a horizontal environment. However, this function only supports three modules of Android mobile phones by the time the implementation was done\(^2\). One of the thesis goals is to develop an AR game, which can reach the most users. The lack of supportive devices makes it an impossible task to accomplish, so Ground Plane was abandoned in the end.

Since it is no longer possible to place the treasure on the ground, it can be placed in the air. To make the treasure semantically correct, a flying bird model is chosen as the treasure demonstration. Animations are implemented on the model so that the bird is spreading its wings, which makes it vivid and realistic in the space. During the game, the bird is set to invisible if no one finds the bird. After someone arriving the right position, the bird is visible to the player. Figure 4.7 shows how the “treasure” bird is existing in the indoor environment after being found out.

4.7.2 Capture Treasure

Finding the bird does not mean capturing the bird. A game is not fun if the player simply walks to a place and wins. A swiping ball function is carried out to increase the playability of the game. The idea is that after the player arrives the position of the treasure, the treasure appears on the players client. Then, the player can fire a ball by swiping the screen. If the ball hits the bird, the player will get the bird and win the game. Figure 4.8 shows the ball just being shot out.

In Unity, a GameObject has to have a Rigidbody component to have physics effect. The fire ball contains a Rigidbody. Then, the detection of the

\(^2\)April, 2018
ball and the bird is given by `OnTriggerEnter(Collider)` function. Both GameObjects need a Collider component. The function is triggered, when one Collider enters the trigger, then Unity will acknowledge the occurrence of the collision. The PlayerController mentioned in 4.6 will change the state of the treasure, therefore the player wins the game.
Chapter 5

Evaluation

This chapter talks about the final evaluation of the game. Section 5.1 talks about the result of game performance evaluation from me. In section 5.2, the feedback results from the user test is discussed.

5.1 Result Evaluation

In section 3.1.2, several fundamental requirements of the thesis were listed. This section talks about whether those requirements are met in the thesis. To summarize them again, they were:

1. Localize the player in the indoor environment.
2. Display treasure using AR technology.
3. Deploy a game data server.
4. Implement multiplayer mode.
5. Reach the widest users.

Requirement 1 is considered as accomplished, since the game client is able to localize the player by using service from VimAI. In section 4.1.1, compressing images does not affect the success rate of localization. Furthermore, the compression method cut down the data size, which in turn reduces the transmission time. The system is able to locate the player in pre-modeling buildings.

Here comes to the evaluation of requirement 2. In this thesis, AR registration is done by indoor localization. Combining with the DEVICE_TRACKING mode. The output treasure is virtually generated in the indoor environment,
which is rendered on Vuforia ARCamera. The tracking part of AR is well fin-
ished by Vuforia engine as well. This requirement is considered as excellently
achieved.

A game data server is built with fulfilled functionality. The client can
store and retrieve the position data via HTTP request successfully. The
server is deployed to the cloud and operates in good condition, so requirement
3 is examined as fully achieved.

Requirement 4 is successfully finished by implementing Unity multiplayer
network interfaces. The game is able to play via multiple players. There are
three various roles for the game devices, a client, a host, or a server. Each
round of game requires one host or server, then other players can connect the
same game as a client. Each game only gets one winner, and the multiplayer
mode approves only one winner could exist.

Requirement 5 is easy to check, since the the game supports Android
devices and runs very well on a mediocre cellphone. It is able to run on most
of the Android phones at present. Developing the game on Unity also means
the capability to build it on iOS devices. In conclusion, the game supports
the majority of mobile equipment.

5.1.1 Performance

A series of tests were done in order to evaluate its performance from software
aspect. During the test, several performance shortages were found.

Game Performance The game was played on the two mobile phones
mentioned before. Unfortunately, one problem happens when playing the
game using Nexus 5, its accelerometer cannot work properly, and thus it is
unable to estimate the pose of the device correctly. It cannot render the
AR content precisely on the screen and the AR effect deviated all the time.
On another device, Huawei Honor 8, the accelerometer works correctly so
the AR function works perfectly as well. But after playing the game for a
period of time, the phone turned to be very laggy. Unity Profiler was used to
analysis the memory cost of the game. The result is shown at figure 5.1. Ac-

cording to the result, it turned out that the game consumes more and more
memory every time it takes a picture. C# has garbage collection method,
which means C# would clean unused object automatically. However, in this
game, C# does not collect unused image objects and they have to be deleted
manually after using. I did not implement it so the image garbage will always
be alive and consuming system memory, which causes the memory overflow.
Figure 5.2 shows the memory performance after implementing garbage col-
lection of the image object. The game will collect the unused images and
thus the memory will not overflow.
The overall game performance is good. Every function works perfectly and smoothly. However, there is noticeable screen froze when the game taking the picture. Figure 5.3 shows the performance info of Hunt scene.

The blue impulse appears when the game takes a photo, it shows the Image.Start() [Coroutine:MoveNext] costs 98% CPU runtime resource at that time. This action happens when Unity takes pictures and sends the image data to the cloud data server.

Battery Performance During the test, the phone turned to be a little hot after playing the game for several minutes. The game requires accelerometer, gyroscope, CPU and GPU to work together, so it is reasonable to consume a lot of battery power in a unit of time. Inspection on energy consumption is then necessary. An app called GSam Battery Monitor was used to test the battery consumption of the game. The game has being played for 30 minutes and then the battery data was collected. Pokémon Go was also executed on the same phone for the same length of time, to make a comparison. The result can be seen from figure 5.4 to 5.7.
Figure 5.4: Total Power Consumption of Thesis Game

Figure 5.5: Total Power Consumption of Pokémon Go
During the 30 minutes game play, the cellphone consumes 559mAh of battery power. Figure 5.4 shows the percentage of each subsystem costs. All the apps together spent 68% of battery power, which is around 380.12mAh. The screen also spent 167.7mAh, which is another main consumption when running a game on a cellphone. Figure 5.6 shows the proportion each app consumes. It shows the game consumes most of the power of application consumption. It costs 57.6% of the power, which is 218.9mAh. After comparing
CHAPTER 5. EVALUATION

these results with figure 5.7 and figure 5.5, which indicates that the game spends a little more battery energy than Pokémon Go.

5.2 User Test

In order to have a comprehensive evaluation about this game, a user test was designed and 12 people in the campus were invited to my place. To begin with the test, they were told with the basic instruction on the game rules. Then, they were asked to play the game with others. After that, they were required to filling a questionnaire, which asked them some questions about the game.

In the question section, each participator was asked whether they feel the game has deficiencies. There are some problems they mentioned in the feedback. The most common problem is that the locating is slow. There are three comments about this issue:

1 “locating is slow”

2 “this game needs some optimization, both localization and performance."

3 “cannot locate most of the time”

The “slow” they mentioned in the questionnaire means they have to send a lot of times of pictures to locate successfully. This issue may happen when the user is in some particular areas, where the environment has changed a lot since last time building the point cloud. Consequently, the server cannot get enough feature points from the taken picture. Another reason may be the picture jitter, which happens when taking picture while walking. From this result, it appears that it is not practical to locate using image-based method all the time.

Except for the locating problem. A user mentioned that the AR effect can be more realistic, he suggested to implement AR with environment understanding function. Another user said the entire user interface can be more attractive. All the improvements can be done will be discussed in section 6.1.

The entire questionnaire is shown in appendix B. In the last two questions, each user was asked to give a overall feedback of the game and the idea. The questions are:

1 From 1 to 10, how much do you like this game?

2 From 1 to 10, how much do you love the combination of AR and indoor localization?
The result of question 1 is shown in figure 5.8 and the result of question 2 is shown in figure 5.9.

Figure 5.8: Answer of question 1

Figure 5.9: Answer of question 2

In the result, it shows that most users gave positive feedback on the game itself. 11 out of 12 users gave 6 or more than 6 points about the game. The average score of this result is 7.3/10. As for the result of their opinion on the combination of AR and indoor positioning, the result indicates that 4 people gave full points and 5 people gave 9 points on that. The final average score is 8.8/10. This results shows that most of the users agree that the
combination of AR and indoor localization is successful and amazing. The limit of this user test is that only university student from age 20 to 28 were invited, which may not be a perfect example to evaluate a success of a mobile game. However, the advice they gave to me is helpful and inspiring.
Chapter 6

Conclusions

The goal of this thesis was to develop an AR treasure hunt game using image-based indoor localization technology, and to evaluate the feasibility of the combination of AR and indoor positioning. The result of this thesis proves that using these technologies can build a Pokémon Go style games. The advantage of indoor positioning allows the game can be played in indoor environment, and hide the treasure more accurate than GPS-based treasure hunt game. Although making a treasure hunt game is difficult with GPS, due to the deviation of the localization service, indoor positioning technology makes it simple. AR also enhances the gameplay of the treasure hunt game. The use of multiplayer mode adds more fun to this game. The final production with those functions proves that the first goal was reached. The positive feedback from the user test also certifies the public feasibility to the mixture of those techniques. Therefore, the goal of this thesis is achieved.

After finishing the thesis, research questions can be answered here. To the first question, “How can we combine the existing AR and indoor positioning technologies?” It shows image-based indoor localization is an ideal method for initializing the position of the device. It runs fast and outputs accurate result. However, according to the feedback from the user test, it is annoying to use image to locate all the times. The better way is using image-based localization at the initial point. Then, combining step detection or other sensory data to track the movement of the device. To answer the second question, “What are the special problem areas when using indoor positioning in AR?” The answer appears to be the descending of localization accuracy, which happens when the cloud point model has not updated for a long time. To avoid this problem, the cloud point model should be constantly updated.
6.1 Future Work

Although the final game meets the fundamental requirements of the original idea, it is far from perfect. In order to deliver a real commercial game for a large group of players, several aspects could be done in the future.

**More Advanced SDK** AR technology is promoting very fast recently. When this thesis was going on, Google released the first stable version of AR SDK - ARCore v1.0. With support to many models of Apple and Android devices, ARCore offers advanced features including motion tracking, environmental understanding. In the user test, some users suggested to add space understanding features in the game. As the mobile phone industry is also growing fast, most of the devices in the future would let ARCore unleash all the power. The usage of AR can rise to another high level.

**Step Detection** The shortage of image-based localization notified me that I should combine the localization with assistance. Step detection can help the game get rid of localization after getting the starting position known. Another shortage of this game is when the AR registration finished, the correlated position of the treasure and the player is settled. It affects the realism of the game if the player moves to another position. Using step detection will collect the accelerometer data from the phone and adjust the position of the treasure continuously.

**User Registration** In order to shorten the development time of the thesis, I dropped some functions which I believe are less-important. User registration is one of them, since the game can identify the player by the network Id. However, a real released game must have a user registration activity, which allows the game add more features on the user. At the same time, a test user asked me to add social network registration function in the user test feedback. In the future development, a user registration and login system are going to be finished.

**Cloud Delivery** Except improving the performance by optimizing the game script, I can also make progress by changing the system architecture. In section 3.2, the game client processes most of the data and functions, then it only sends the position data to two servers. A recently emerging system architecture allows the client only capture the sensor data of the device and send all the data to the cloud platform, which is built by AR streaming framework. AR content rendering consumes most CPU resources. It is much efficient to render the AR content at cloud and transfer the stream to the client. A cloud based AR streaming platform can save a lot of memory and battery resources of user clients.

**Performance Optimization** In section 5.1.1, causes of performance
problems were discussed. I will modify the C# scripts and optimize the game structure to promote the game performance.

6.2 Final Thought

Both AR and indoor positioning techniques are in the ascendant. However, they have already shown the great benefits they can give to the mobile applications. This thesis is one example to show the power of the combination of two techniques. I hope the work I have done can inspire others to explore the application of AR and indoor localization. I also expect there will be more successful use cases in the coming future.
Bibliography


[34] WIKIPEDIA. Euclidean distance — Wikipedia, the free encyclopedia, 2018. [Online; accessed 4-June-2018].


Appendix A

Response Data Format

The example JSON response data from VimAI localization Server

```json
{
    "position": {
        "x": -1.774667779333985,
        "y": 1.533781484972505,
        "z": -1.3345878483144855
    },
    "direction": {
        "x": 0.01033899985647412,
        "y": -0.9997907864468742,
        "z": 0.01554096977156727
    },
    "rotation": {
        "x": 0.46243149042129517,
        "y": -0.5251068472862244,
        "z": 0.5379886031150818,
        "w": 0.4700937569141388
    },
    "areaID": 2,
    "timestamp": 1529418861064,
    "statistics": "3D2D: 46; inliers: 24; threshold: 5.73 539; focal: 1002.06; ppoint_x: 312.606; ppoint_y: 6 38.519; IRMSE: 1.58228; FRMSE: 1.11601; residuals: 48; estimated_focal: 1066.67; partition: 1;"
}
```
Appendix B

Questionnaire

1 What is your age?

2 What is your gender?

3 Do you know what Pokémon Go is?
   □ No, I've never heard of it.
   □ Yes I have heard of it, but I have little knowledge about it.
   □ Yes I know how it works.
   □ I installed it in my phone.
   □ Others.

4 Do you like playing treasure hunt game with more accurate localization?
   □ Yes.
   □ No.
   □ I am not sure.

5 The following questions are based on the thesis game. What do you feel good about this game?
   □ The game strategy is novel.
   □ The localization method is amazing.
   □ The multiplayer mode is good.
   □ Others, please specify.

6 If this game come to commercial, will you like to download it?
☐ Yes.
☐ No.
☐ I am not sure.

7 Do you think this game has any problem?

☐ Yes.
☐ No.

8 If you think this game has problems, please write them down.

9 From 1 to 10, how much do you like this game?

10 From 1 to 10, how much do you love the combination of AR and indoor localization?