CROSS-PLATFORM INDOOR NAVIGATION USING MIXED-REALITY

by

Akshay Ayyanchira

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Computer Science

Charlotte

2019

Approved by:

____

Dr. Aidong Lu

____

Dr. Mohammed Shehab

____

Dr. Min Shin

ABSTRACT

AKSHAY AYYANCHIRA. Cross-Platform Indoor Navigation using Mixed-Reality.
(Under the direction of DR. AIDONG LU)

Indoor navigation and localization are challenging due to the limits of available devices. A number of devices, such as GPS [3], beacons [5], infrared positioning system, ultrasonic positioning system and RF positioning system [21] have been explored. We propose to utilize the latest mixed-reality technologies to achieve indoor localization and navigation functions.

In this thesis, we propose a cross-platform system for navigating indoor environments using Microsoft HoloLens and iOS devices. Our approach leverages on device sensors and Vuforia API of image marker to localize users inside a building. The user locations are further synchronized among all devices connected through the interface of Unity networking, which is scalable and reusable for both HoloLens and iOS platforms. We provide both individual view version and server version for different purposes, both visualizing user locations in real-time when users are connected through HoloLens or iOS devices. We develop several different designs for visualizing 3D building model, so that users can choose them according to their exploration tasks. While isometric visualization provides complete structural information of building in 3D, staircase visualization reduces occlusions from isometric view by arranging the floors in a staircase fashion. Additionally, orthographic visualization provides top view of all the floors. We also add an immersive view which enhances navigation by providing augmented visualization for mobile devices.

We have conducted a set of small user studies covering different use cases of the project including building navigation, friend finder and rescue operation. Users were given different devices and were asked to switch between different visualizations as they performed the tasks. We also obtained expert feedback from a psychology professor. Results have demonstrated promising results for this technique for everyday building navigation and emergency situations.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Aidong Lu, and my committee members Dr. Mohammed Shehab and Dr. Min Shin, for giving me such a great opportunity to work on this project. I would also like to thank my colleagues Wilis Fulmer, Tahir Mahmood for their active participation in discussion and decision making while the core component was being developed. Special thanks to Elias Mahfoud for such an inspirational project and guidance along the path.

This thesis is based upon work supported by UNC Charlotte and the National Science Foundation under Grant Nos. 1629913 and 1840080.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

With its first fully operational unit in 1995, GPS has since evolved tremendously and found its place in mobile phone, cars, camera, and many other gadgets. Location services have many benefits in fields like navigation, healthcare, marketing, weather forecast etc. With limited GPS access inside building, we use Vuforia Image Recognition to locate users inside the building. Using Augmented Reality (AR) in HoloLens and iOS, we push our capabilities further by helping users understand their locations in 3D interactive building models. The release of Microsoft HoloLens has been a breakthrough in the world of mixed reality. Combining the field of AR and Virtual Reality (VR), mixed reality (MR) devices have added functionality of spatial perception which allows the real world to be blended with virtual objects. In this thesis, we allow users to use Microsoft HoloLens and iOS devices for building exploration and navigation purposes. With processors and sensors with similar performances, tracking movement of users carrying an iPhone or iPad is comparable to those wearing HoloLens.

Chapter 2 describes related work in the field of AR and VR. Different researchers have explored different approaches to visualize 3D structures. Some work has collaborated settings with different devices like HMDs, flat screen displays, and cave style environments; while others have focused on individual project such as multi-dimensional data exploration. A study of these papers has been useful to understand capabilities as well as drawbacks of AR and VR system.

Chapter 3 introduces terminologies required to understand the implementation of core components. It also covers the basics of Unity IDE. Maintaining hierarchical structure of building floors and player object inside the project is a part of location tracking and hence it is important to understand this Unity project layout. This chapter also elaborates networking architecture and image recognition libraries used in the thesis.

Chapter 4 focuses on implementation of core system component. It explains project architecture and initial configuration required for the project. This includes setting up Unity, arranging building floors, creation of different projects and using camera component for movement tracking. Image registration and the technique of finding user's location using Vuforia Image Recognition is explained in this chapter. The flow is once again explained using help of an example.

Chapter 5 elaborates on custom navigation system. This includes five stages of operations for calculating paths from source to destination, consisting graph generation, destination selection, collection of paths, shortest path selection and path visualization. It also covers the different type of interaction provided based on device type. Three visualizations for exploring the building are also discussed in this chapter. They enhance the location awareness among the users. Users get a glimpse of overall building structure using the isometric view which is launched by default. Though the models have transparent materials which allow them to see through walls, staircase view arranges the floor distinctly allowing user to have clear view on each floor with no occlusion. Orthographic view helps understanding precise location from top view as it gives distance information of users relative to walls. Leveraging interface guidelines of respective devices, we enable multi-touch interaction on phones with features to pan over the model and pinch to zoom. For HoloLens, we use voice commands like “Follow” and “Stay” to make the model anchor to a 3D place in space or follow the user while they walk inside the building. We have also incorporated gesture detection and used one and two hand manipulation to re-position, rotate and scale the model. This chapter also covers the AR based immersive view implemented for mobile devices.

In chapter 6, we discuss 4 different tasks involved in user study. Tasks 1 and 2 were for individual users on navigating to specific destination using three visualizations or AR immersive view. Tasks 3 and 4 were for collaboration, where users could see each other’s location and performed joint tasks. We have also acquired expert feedback on our system to understand human behaviour and the psychological aspect of this project.

In chapter 7, we briefly go through the results of the user study and conclude key benefits and drawbacks of this thesis. Future work of adding new features like www networking along with suggested improvements are discussed here. With many devices being AR-enabled, this thesis serves as a reusable core component for many applications, where connected components are in same shared environment.

CHAPTER 2: RELATED WORK

2.1 Immersive Analytics

Immersive analytics [6] extend the classical desktop visualization into a variety of new environments, including AR and VR. While still in its early stages, immersive analytics has attracted the interests of many researchers. Both prototype systems of immersive analytics systems have been developed utilizing the 3D virtual or physical space to explore different data tasks and interactions, and evaluation studies have been performed on the effectiveness of these approaches [16].

AR superimposes holograms with the environment around users and allows interaction with holograms and everyday objects together. The Microsoft HoloLens is a common example [7]. AR is more suitable for real action with the integration of virtual information in real physical environments. For example, AR was used as a tool to support the collaboration between rescue services for the police and military personnel in a crisis management scenario [20]. AR techniques were used to support quick context-related information exchange for operational units in the security domain that work together in teams [11]. AR-based mobile assistance systems in context-based provision of facility-related information [17] were shown to minimize the intensive recall required in this domain. Mahfoud et al. [18] presented an immersive visualization approach for investigating abnormal events in heterogeneous, multi-source, and time-series sensor data. Tahir et al. [19] explored AR for visualizing bio-diversity data in a large physical environment. Recently, a toolkit for building immersive data visualizations based on the Unity development platform has been published [22].

VR artificially creates sensory experiences, which may include sight, hearing, touch, and smell. Generally, VR is used for training purposes and it allows users to experiment in real time under various situations such as evacuation scenarios [23]. Immersive analytics techniques in VR often take advantage of the advanced rendering and storage capabilities. Kwon et al. [15] investigated the effectiveness of graph visualization and the impact of different layout techniques on readability in an HMD, and they concluded that the 3D stereoscopic graph visualization using Oculus Rift out-performed traditional 2D graph visualizations. Usher et al. [25] developed a VR system for users to trace 3D neuron structures from high-resolution images. Cordeil et al. [8] presented ImAxes system which allowed users to manipulate 3D axes like physical objects and combine them into sophisticated visualization for exploring multivariate data. Yang et al. [29, 30] explored different ways to render world-wide geographic maps and studied the origin-destination flow data in a global geographic context.

Specifically related to immersive analytics, a number of recent studies have been performed and the results are still mixed. For example, studies of performance on collaborative immersive visualization using the recent HMDs, such as Oculus Rift and HTC Vive, have shown no difference with expensive equipment such as cave-style environments [9]; while Kwon et al. [15] concluded that the 3D stereoscopic graph visualization with an Oculus Rift out-performed traditional 2D graph visualizations. Also, Bach et al. [1] showed that both desktop and immersive environments are more effective for certain tasks, but that generally the desktop environment was still fastest and most precise in almost all cases. Recently, immersive navigation [28] showed significant performance improvement in VR and AR environments, on tasks that include tracking, matching, searching, and ambushing objects of interest. Overall, Some advantages of immersive analytics have been demonstrated, especially favorable results for stereoscopic techniques [9, 15, 27].

This work presents an AR system, requiring us to handle issues of networking between online servers and AR devices ,registration of AR devices with the real world and navigation inside the building.

2.2 Cross-Device Techniques

AR developers face a proliferation of new platforms, devices, and frameworks, leading to new applications and techniques created with cross-device approaches. Similarly, ubiquitous analysis integrating different devices and frameworks were also proposed [12]. For example, the GraSp approach [14] demonstrated that spatially-aware mobile displays and a large display wall can be coupled to support graph visualization and interaction. Horak et al [13] presented the combination of smartwatches and a large interactive display to support visual data analysis. Butscher et al. [4] combined a touch-sensitive tabletop and AR headsets to visualize clusters, trends, and outliers in multidimensional data. Recently, Speicher et al [24] developed a taxonomy of AR system components and identified key challenges and opportunities in making them work together through a review of existing AR platforms and a survey of 30 AR designers, developers, and users.

Cross-device techniques often need to handle the networking and communication among multiple devices. For example, VisHive [10] constructed web-based visualization applications that can transparently connect multiple devices. Similarly, our approach contains a client-server networking component to stream data from server to connected clients and vice versa on press of a button.

CHAPTER 3: TECHNICAL BACKGROUND

Because we are trying to implement multi-platform networking, this thesis focuses on using Unity to develop projects for iOS as well as Microsoft HoloLens which enables seamless switching of platforms. An alternative to core functionality, though not tested, should be portable to iOS using native development workflow like SceneKit with ARkit in Xcode for iOS devices instead of using Vuforia SDK on Unity. Also, because HoloLens requires different assets than iPhone/iPad, different projects have been created for different platforms to keep the code isolated.

3.1 Unity Development Environment

Unity 5.2 was a major release which replaced many core functionality of Unity. One of the biggest update was Unity 5's new Networking System. Since then, Unity has improved and made the networking system more robust. They also made project (Tanks Multiplayer) available in its asset store to get used to the modern way of networking and lobby system where players can connect and get ready for a session to start. Though, the documentation to use the system is not out yet, a video was published in Unite conference showcasing how to use the new networking system. Today it has become easier to create a multi-player game/project. Network Manager and Network Identity scripts attached to each gameobject(a player or any shared asset within connected players) makes it possible to add many players and objects in a connected session. By using NetworkTransform component, a player can also track other player's 3 direction movement. These reusable assets are plug and play and makes the process very easy. However, all these benefits only works if the build is made from the same project. Though it was easy to build the same game for HoloLens, iPad and Desktop and make them talk to each other, we couldn't create two different projects with similar assets and make them interact with each other. Internally, Unity provides each asset a unique serial number and uses that while spawning the game object. Keeping different projects in Unity compromises the ready-to-use multiplayer networking capabilities that Unity provides by default. Using the same project is not ideal because each device will have its own custom UI, different requirements and in future, cannot be guaranteed to be supported by Unity. Hence, we try to make the coordinate systems work flawlessly without the using the Network Transform component of the Unity.

To understand the implementation of this project, it is required to have understanding of basic terminologies in Unity.

3.1.1 Scene

A scene, simply put, is an environment where we have our world setup. Just like how a website has an starting html page containing all the objects for that one page, a scene has all the objects called as 'GameObjects' arranged in a 3D space which loads when an application starts. Unity allows you to choose either 2D or 3D space, but because we are focused on creating Mixed Reality application, we choose 3D space. Unlike many programming languages and platforms with one entry point main function which then calls other functions, in Unity the project starts by loading a scene.

3.1.2 Game Object

It is one of the most important concept and what makes Unity special. Every object inside a scene or your game is a GameObject. It can be a light source, a player, ground, walls, buildings, or even non-visible things like GameManager and NetworkManager which only do logical operations. If you are familiar with iOS development, a gameobject is analogous to a SCNNode in SceneKit. In the next chapters, we talk about keeping a gameobject as a parent and child in our explanation. Hence it is necessary to understand that in Unity, a gameobject can have another gameobject in it. Each Gameobject has a transform component, which represents its location, rotation and scale in a 3D space.

3.1.3 Hierarchy

There is a local space and a world space for each gameobject. If a gameobject does not have any parent, the local space is itself the world space. However, if a gameobject is a child of some gameobject, its transform is set with respect to its parent. In this case the local space and the world space differs. This can be easily understood with the diagrams used in implementation section.

3.2 Vuforia Image Recognition Library

Vuforia is one of the core SDK that is used in the project. Vuforia enables us to build Augmented Reality Application for devices ranging from iOS and Android mobile phones to wearables like Microsoft HoloLens. Its SDKs can be integrated with Xcode, Unity, Android Studio and Visual Studio. Vuforia SDK uses the device camera and scans for specific image patterns called 'Image Targets'. On detection, the program invokes a function with

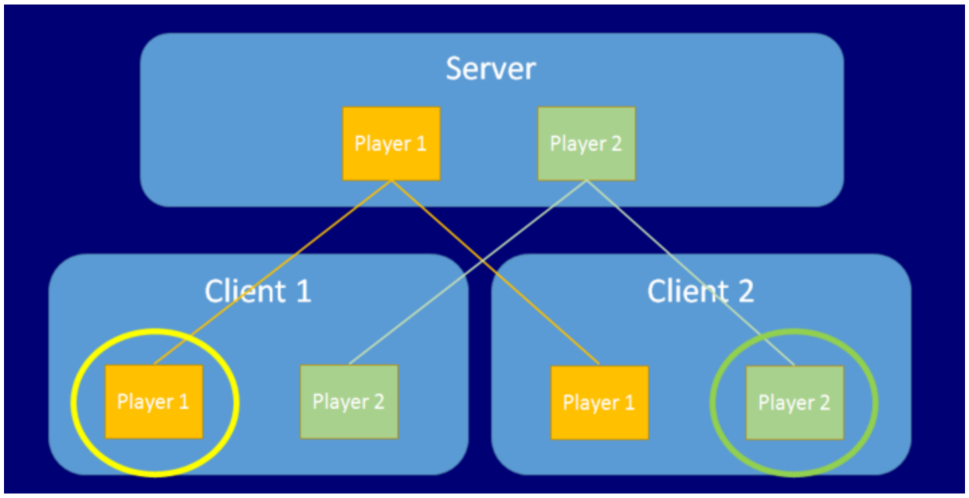


Figure 1: Unet Networking Architecture

the information of the image target that has been detected which includes its name, distance, rotation and scale. In our project, Vuforia plays its part to identify image targets which is placed at different places on the floor of the building.

Technically, the purpose of Vuforia can also be fulfilled by using device native frameworks like ARKit for iOS and ARCore for android, but because Vuforia runs cross-platform, we chose Vuforia so that the code for post-image-target detection methods can be re-used.

3.3 Networking Components

Unet networking offers two primary modes to establish networked projects.

1. Peer to peer based projects and
2. Single server.

In peer to peer based project, the first user to start the application becomes the host, which acts as a server and a client. Usually when this client disconnects, the server role is passed to another user. However, in single server mode, we have a machine dedicated to do server specific tasks. Unless the server is ready, no user can connect to a session.

The image above, referenced from Unity's docmentation, demonstrates how two clients connect with the server and have their properties reflected on other client instances. The single server approach is used in this project to keep the components isolated and modular.

Unet communicates with connected components using MessageBase derived classes. Message classes can contain members that are basic types, structs, and arrays, including most of the common Unity Engine types (such as Vector3). They cannot contain members that are complex classes or generic containers. The project uses two such custom classes inherited from MessageBase class, one for client to server and other for server to broadcast device positions of 10 connected clients. The MessageBase derived classes can be introduced to add new data into the architecture.

3.4 Spatial Anchors and Synchronization

The device movement and image recognition is carried out using SDKs provided by the device manufacturers as well as from SDKs from third party companies. Few of the libraries referenced are described below.

3.4.1 Microsoft Sharing Library

Microsoft announced the world's first independent mixed reality device on March 2016 - The HoloLens. This device runs on Windows 10 and has array of cameras and sensors for spatial understanding of the user's space and displays Holographic content which is interactable to touch and sound. Microsoft released a collection of document and tutorial videos for developers to get started with device called Microsoft Reality Academy. The tutorial demonstrates how to use 'HoloToolkit', a package provided by Microsoft to readily integrate with new Mixed Reality Projects. They have a ready to use library called the

’Sharing’.

The HoloToolkit.Sharing library allows applications to span multiple devices, and enables holographic collaboration. Originally developed for OnSight, a collaboration between SOTA (a Microsoft studio) and NASA to enhance their existing Mars rover planning tool with HoloLens, HoloToolkit.Sharing enables users to use multiple devices for a task by allowing the apps running on each device to communicate and stay in sync seamlessly in real time. Users can also collaborate with other users (who are also using multiple devices) who may be in the same room or working remotely.

3.4.2 Apple ARKit 2

ARKit 2, the second version of AR platform by Apple was released in June 2018. Being a mobile library, ARKit became world’s largest AR platform as its competition ARCore was in its making but is now available with equal potential. ARKit 2 allows developers to integrate shared experiences, ‘persistent-AR-experiences’ tied to a specific location, object detection and image tracking to make AR apps even more dynamic. Shared experiences with ARKit 2 make AR even more engaging on iPhone and iPad, allowing multiple users to play a game or collaborate on projects like home renovations. Developers can also add a spectator mode, giving friends and family the best views of AR gameplay from a different iOS device. Persistent AR will also change the way consumers interact with AR apps by creating opportunities to leave virtual objects in the real world to which users can return. They can start a puzzle on a table and come back to it later in the same state or create an art project over the course of a few weeks without starting over each time. A good example was given during the WWDC event where the game of LEGO was played and saved in virtual environment.

3.4.3 Google AR-Core

Unlike Apple's ARKit, Google ARCore provides similar functionalities but not just limited to Android but also to other development platforms like iOS, Java, Unity(iOS and Android) and Unreal. Users in the same environment can add the Cloud Anchors to the AR scene that they see on their device. Your app can render 3D objects attached to the Cloud Anchors, letting users see and interact with the objects simultaneously.

To make these shared AR experiences possible, the ARCore SDK uses Google servers to host and resolve anchors.

3.4.4 Mixed-Reality Squad-Coordination Platform

Developed by a former student from UNC-Charlotte, the Mixed-Reality Squad Coordination Platform assists the coordination of squads when performing tasks in real time. The platform provides tools to deliver visualization and heads up display (HUD) information to squad members, as well as reporting tools that allow squad members to coordinate with each other and their command center. The platform uses HoloLens to synchronize member locations and other vital information across the team. Such a system allows dispatchers to efficiently exchange useful visual information, such as targets and paths with field units.

CHAPTER 4: SYSTEM

Devices are evolving and each device is unique in its own way. Google Glass was one of the early devices which could project virtual objects in front of us using a small prism near the right eye. Microsoft HoloLens, added arrays of camera, sensors, infrareds, speakers and powerful processor and a battery to provide wider, more realistic holographic views with hand gesture detection. It also added spatial perception which makes the device extremely powerful to suit many modern use cases. HTC Vive, Oculus Rift and other VR devices also proved their capabilities by showing the immersiveness they can provide. Recently announced Magic leap adds controller support and has a seperate cabled processing unit to keep the main headgear light weight. Each device comes with its own toolkit for development. It is important to notice that the SDK specifies which Unity version is it compatible with. Though Unity is usually backward compatible, for HoloLens development, it is recommended to use the Unity version specified in Holotoolkit documentation. During this project development, the project for HoloLens is developed in Unity 2017.1.4 while for Server and iOS, it is 2018.2.

4.1 Project Architecture

This is what the Project Architecture looks like. The project will have a central server gathering information about its connected clients and distributing the information to them. The sole purpose of the server is to keep the consolidated data at one place and broadcast

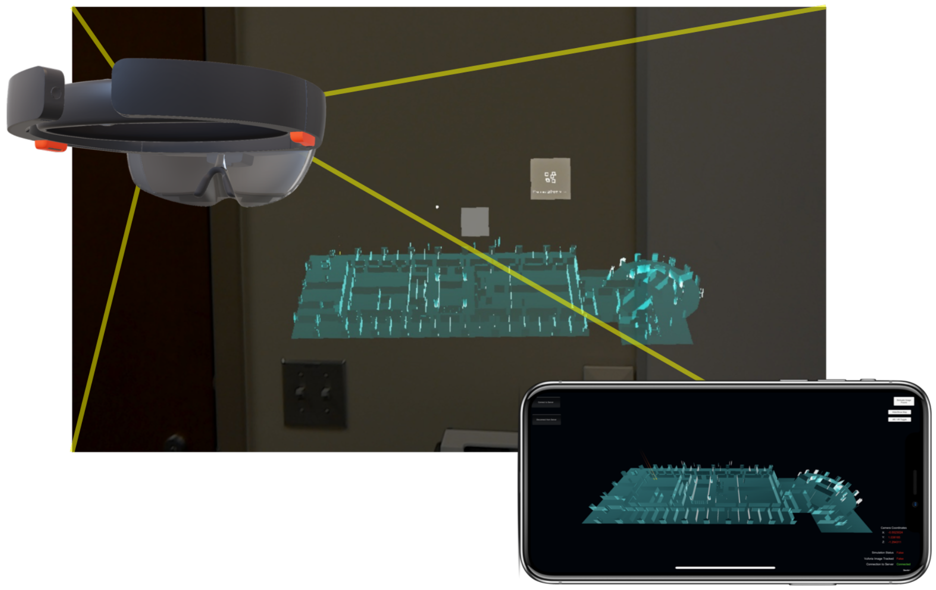


Figure 2: Application Design for HoloLens and iOS device

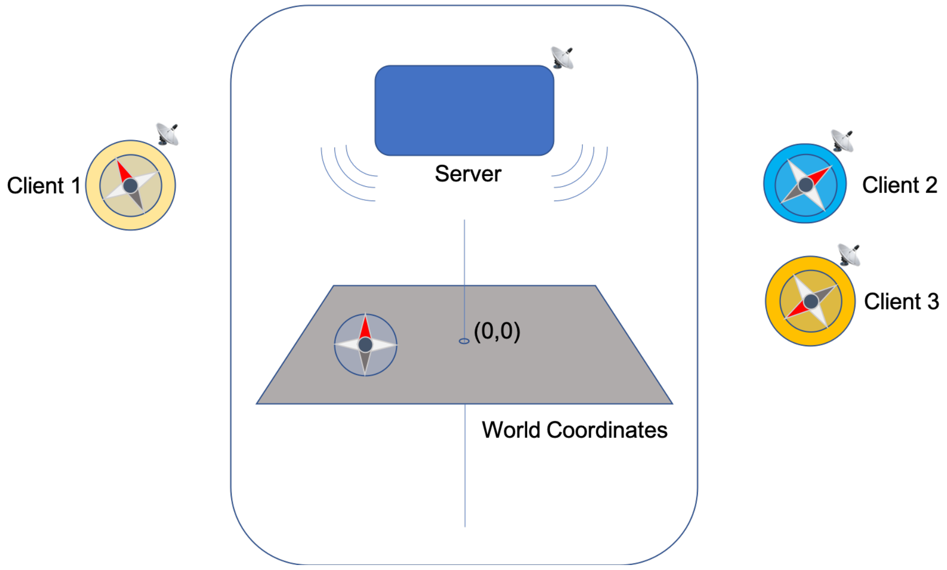


Figure 3: Project Architecture

the data.

The image represents the server with world map which each player has to conform to. Though each player may have its own coordinate system,they are aware of how its translation to server coordinate system will look like once the image target is detected.

4.2 Initial Configuration

When each device starts the application, its coordinate system starts at (0,0,0) world location.

SDKs that come out with these devices have their own toolkits which eliminates the need to develop basic input-output system like Camera, touch input for mobile, gesture or controller input system etc. They are available in the form of prefabs or scripts which is drag and drop enabled in Unity. Vuforia also comes with its own set of prefabs and scripts. One of such component is the 'ARCamera'. By replacing the default camera in the scene, the ARCamera gameobject now serves as the Main Camera which also acts as a viewfinder on the iOS devices and HoloLens. However, to integrate Holo-toolkit (a package used to develop HoloLens projects), the HoloLens Project has two Camera setup. One provided by Holotoolkit, for basic gesture input operations and Vuforia ARCamera for target image detection.

4.2.1 Setting up the scene

This project demonstrates how users in one coordinate system can communicate with users in other coordinate system when both comply to same universal coordinates. Also, as mentioned earlier, this thesis focuses on creating different projects for different platform to showcase how independent we can get without the network ID based transform syncing.

Because the aim of this project is to have different devices to communicate their coordinate system which is universal, it is important to have the same scene arrangement across all the projects. This means that a building model with its origin placed at (0,0,0) world coordinates on server's scene should be similarly placed in other project's scene.

4.2.2 Player and Image Target Setup

Have a player prefab ready in the scene which will be used by network manager to spawn it when it connects with server.

Vuforia SDK is used for detecting image targets throughout the floor area. For maintaining the consistency, each project should have image targets to be placed in exact same locations in their scenes. For e.g, if a image target is placed near the lobby's door in project one, it should be placed near the lobby at exactly same distance from the door as in the scene for project 2. Also, the image target prefab by Vuforia is laid flat on the ground. In our project, as the image will be sticked on the wall, the image target is rotated on x axis by 90 degrees.

4.3 Implementation

The components in the project are interdependent. This means that PlayerMovement Manager, a component which controls the player's location, updates the player's transform based on current image detected and the movement, is independent on Network Manager, which streams data to and from the server. Just like modern multiplayer gaming approach where connected players wait in the lobby area, the players with no image target detected are kept near the origin for server to understand that no image target has been detected by those players. Only if a image is detected, their positions are calculated and broadcasted (elaborated later).

Each connected client has a universal map loaded into the project. The map is a 3D model created in Blender and exported to Unity in .obj file format. The same map will be used while displaying on the screen. However, the client will not understand where in the map its location is unless it finds some of the image target and then performs the calculations to determine its location in the universal map.

When a user launches an application, may it be on an iPhone or Microsoft HoloLens, the device's current location and direction is taken as origin. In Unity, the player's transform location as well as rotation is 0,0,0 at that instance. This can be well understood by following diagram.

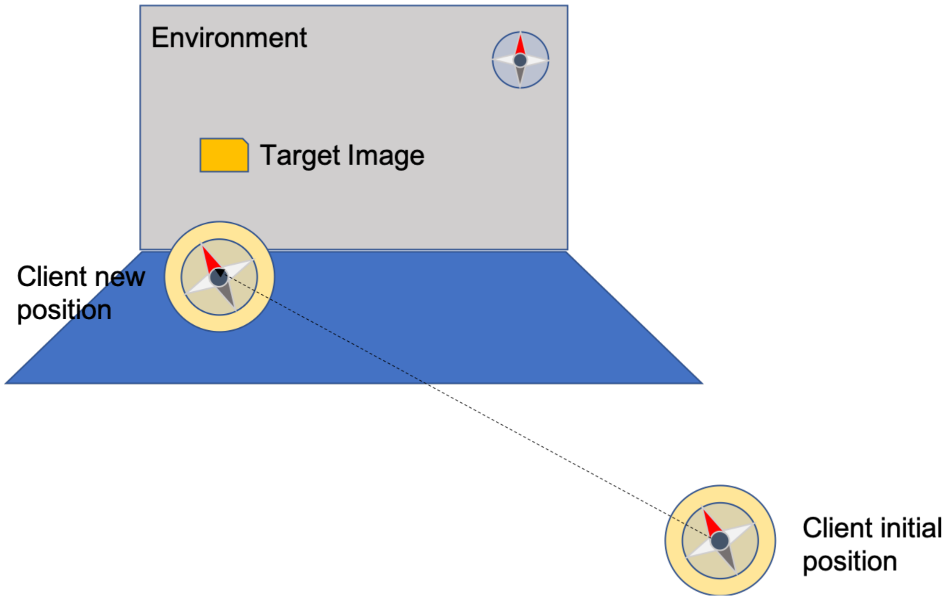


Figure 4: Application Launch

The diagram shows the coordinate system of the environment and the client using the compass logos. The gray wall and blue floor indicate real world with a target image attached to the wall. This also resembles how the universal map stored in each client and the server looks like. It also demonstrates real world placement of image target on corresponding walls. When application starts, the player initializes a gameobject representing itself at its location which is at origin and updates it as the player moves and rotates. Though the player may move forward, it may be diagonal movement in the real environment as shown in the diagram. This gameobject is kept private and not exposed.

Along with the universal map, each client also holds the name, position and rotation of each image targets offline. The universal map also holds gameobjects corresponding to each image targets with its name assigned as gameobject's tagname.

Once the image target is detected, Vuforia SDK returns the distance, rotation and scale of the image as a Gameobject transform along with image marker name which uniquely identifies the image target. This is matched with list of image targets that the client holds offline.

This image target transform and player's transform indicating its current location from origin is then used to create a transformation matrix. Unity has provided a method TRS - which takes in (Translation, Rotation, Scale) as parameters and returns a transform 4x4 matrix. The translation passed is the distance between image target detected and the player's location. The rotation passed is the rotation of the image target. The scale is hard-coded to 1. The matrix returned is inversed and its translation is fetched. This gives us how far and where the player is, with respect to image target in universal map. The client uses this information to create its own replica in universal map by creating a gameobject, with transform values calculated using above approach, as a child of image target.

Once the replica is created and kept as a child of image target gameobject, it has its own local space and a world space coordinates which are different from each other. The local

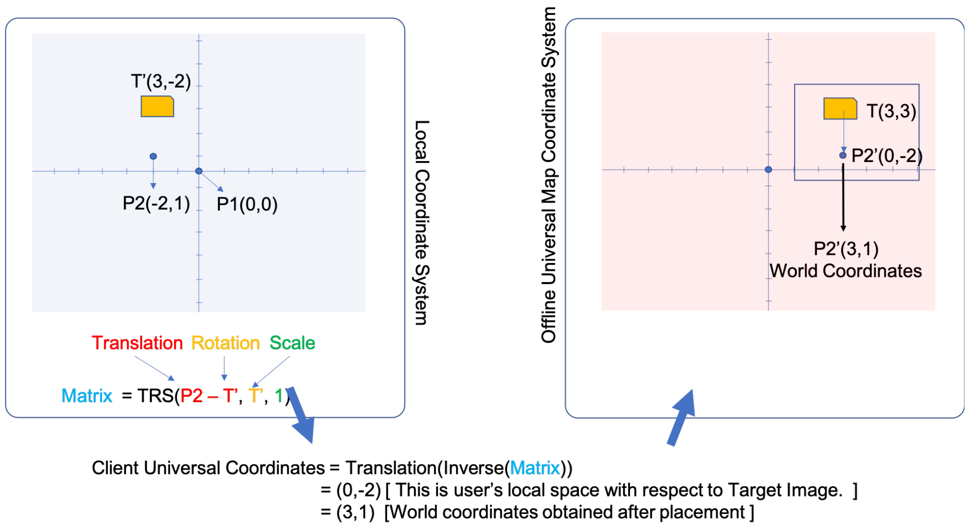

T'(3,-2)
P2(-2,1)
P1(0,0)
Local Coordinate System
Translation Rotation Scale
Matrix = TRS(P2 – T', T', 1)
Offline Universal Map Coordinate System
T(3,3)
P2'(0,-2)
P2'(3,1)
World Coordinates
Client Universal Coordinates = Translation(Inverse(Matrix))
= (0,-2) [This is user's local space with respect to Target Image.]
= (3,1) [World coordinates obtained after placement]

Figure 5: Location Translation with example

transform of the replica is the one we set previously when it was been initiated as a child. However, the global transform of the gameobject will reveal the exact position and rotation in universal world coordinates.

A better explanation can be given using an example. The figure shows two coordinate systems locally stored in the client. All metrics are in meters. For example purpose, we are considering 2D system for better understanding. The exact calculations are used in 3D space.

The blue coordinate system is the one which gets initiated when the application starts and the pink coordinate system is the universal map stored locally on the client device.

$(P1)$ is the initial location of the user when the application starts which is $(0,0)$. and $(P2)$ is the updated location when user moves certain distance.

Lets assume that the image target (T), which is placed at $(3,3)$ in universal map appears right in front of the user at distance of 1 meter when user is at Point $(P2)$. By identifying the unique image target, the application creates a replica of image target(T') at (3,-2).

A transformation matrix is then created where

$Matrix(tm) = Inverse(m)$ and $Coordinates = Translation(tm)$.

$Matrix(m) = TRS(P2.postion - T'.position, T'.rotation, 1)$

$ClientCoordinate = Translation(tm)$

The translation function deduces the distance of the player from the image target in the universal map. In above example, it returns $(0, -2)$. A player's replica is then created (or updated, if already exists) as a child of target image in the universal map stored offline as represented by the $(P2')$. As a child object of (T), Point $(P2')$ has different local and global coordinates.

Finally, Unity allows us to take global coordinates of the point $(P2')$, which is $(3, 1)$, which denotes where the user is in the universal map.

This process is repeated more than 30 times in a second and the coordinates are calculated at a very high precision upto 8 decimals. The distance between P1 and P2 is therefore, very small. This allows precise movement tracking. However, for easy understanding, we considered integers in our example where $(P1)$ and $(P2)$ are more than a meter away.

CHAPTER 5: BUILDING NAVIGATION

Having the map with user in definitely useful. But it won't help much unless the user knows which direction to go, or, if the route he decided is optimal. Hence, we built a navigation system and incorporated with this project. This module helps user find best path to reach to a destination.

5.1 Path Finding

Unity has a strong AI system called Navmesh, which can understand floors, obstacles and is capable to generate paths from one location to a specific destination. This is usually used in games where for example zombies are required to follow the player and attack them. Though this system could help us generate path automatically, we decided to develop our own navigation system as in future, we can have finer control over paths that we obtain and also to collaborate the routes with connected users.

The navigation system is built using node-points spread across entire building. Each node point represents either a turn point, or a point of interest which can be anything from a Professor's office, Classrooms, Meeting room, Computer Lab, Restrooms etc. A node point is a gameobject in Unity's hierarchy, placed inside model of the building as shown in the diagram.

A path between two nodes is calculated using a custom version of Breadth First Search Algorithm. There are 5 stages involved in navigation system:

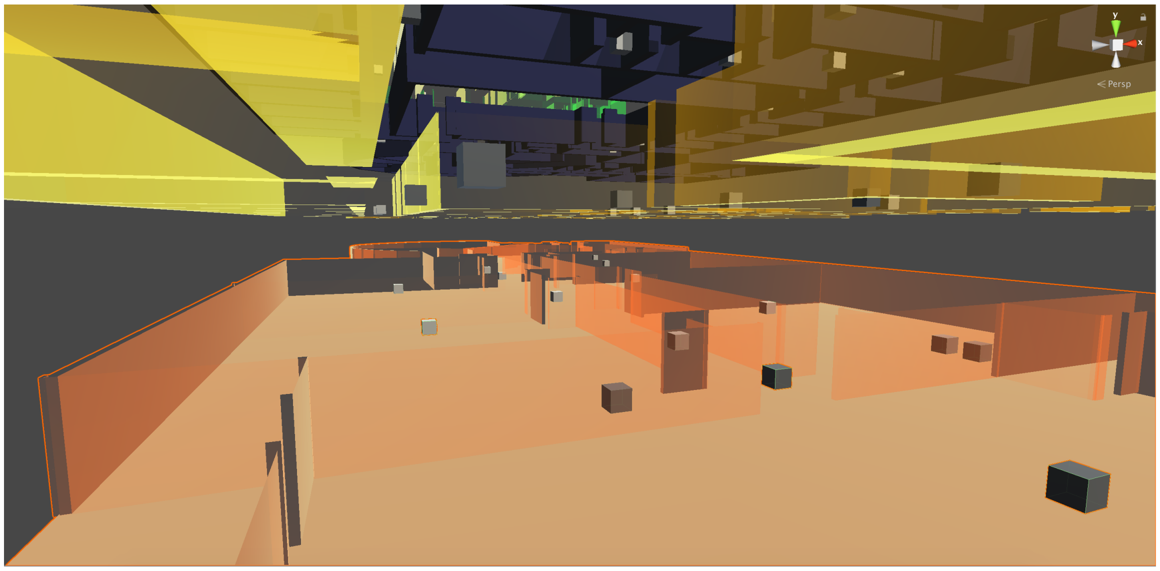
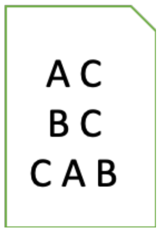


Figure 6: Adding Node Points in 3D Building

5.1.1 Node Graph Generation

To make this application re-usable for new building models, we came up with a format to generate the node graph. Each building model will have its own Graph file, which is a text file which says denoting reachability from each node. For e.g if there are three nodes in a building A, B and C, there will be three lines in the text file:



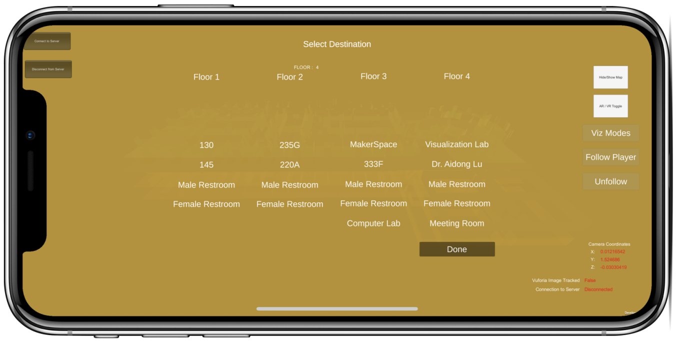
```
A C
B C
C A B
```

Figure 7: Text file for graph generation

The first letter of each line denotes the starting node and rest of the character on the line tells if it is reachable from the starting node. Here C is reachable from A, and also B is reachable from C. A and B cannot reach each other directly. This file is used to generate a graph.

5.1.2 Source And Destination Selection

The project uses a floor based start point. Apart from the graph text file, each building that the project uses is required to have all possible point of interest/turn points for each floor. It is just a text file with point of interest on each line. Precaution is to be taken that the node points inside the building model to have same name as mentioned in the file. The floor is determined based on its y-axis of user's location. The player object is placed inside the floor model based on which floor the user is. Once the floor is finalized, distance from user to each point of interest/ turn point on that floor is calculated and the nearest one is selected as the starting node.



Camera Coordinates
X: 0.01216542
Y: 1.524686
Z: -0.03030419
Vuforia Image Tracked False
Connection to Server Disconnected

Figure 8: Setting Destination on iOS Device

For Destination, we display all the point of interest categorized by floor. As user taps/

selects a location, it is stored as the destination. Selection of a destination is explained more Interaction chapter.

5.1.3 Collecting All Paths

Once the graph is generated, it becomes quite easy to find the shortest path and use it. But considering long term, project stores all the possible paths. A custom version of Breadth First Search Algorithm is used to find all possible paths from start node to the destination. The collection of all possible paths is kept global for visualization module to decide which path to select and present to user.

5.1.4 Shortest Path Selection

For current implementation, the application uses the shortest path by default. As there is no weight defined in the text, it will be unfair to declare ‘more the nodes, longer the distance’ because there can be two nodes really far from each other with no obstacles in between and also two close points with many points in between. To make the shortest path give effective result, the application scans each possible path generated by step 3. For each path, it finds the vector distance starting node to next node, reaching till the destination and stores the path with shortest total distance.

5.1.5 Path Visualization

The visualization of path is achieved using Unity’s Line Renderer component. An object of line renderer is kept global and reused for each new path drawn. The line renderer component requires number of nodes it is going to visit and Vector 3 locations of each node in 3D space. We use the node points created inside the building model for this purpose. There are functions designed to clear the path, redraw the path which is frequently used while switching back and forth from different visualization.

5.2 Interaction

Design plays a major role for user experience. While an iOS App developer tend to incorporate Human Interface Design guidelines in the project, the same is often not applied in creating games or applications having finer buttons and different alignments. For this project, we did not just have two different form factors, but two completely different device with different OS, but both of them wanting to show the same data in a meaningful way. Hence for destination selection and building manipulation, the project implemented different approach for iOS and HoloLens.

Showing destination list was accomplished using basic UI components like buttons, placed in Unity's canvas. The canvas was displayed and hid when navigation button was clicked. The canvas system serves the purpose for 2D flat screen in all the cases. However, this approach is not suitable for HoloLens. Due to its cursor being controlled by head direction, guidelines by Microsoft says that user interaction buttons to be big and within reach of user. It provide visual cues that when cursor is over the button and should have a 3D effect of it being pressed. Thankfully, Holotoolkit, a package contains some reusable components and prefabs that helped in this. We laid out buttons with components like BillBoard, Sphere TagAlong and a custom Animator script to keep it near the user, facing towards the user at all times. The images below demonstrates how buttons are displayed in HoloLens and in iOS device.

Building model was also controlled differently on these devices. The iOS application had specific, but defined visualization of floor models and on the other hand, the HoloLens had only one approach of showing the building model (more elaborated in Visualization module below). While iOS application incorporated multi-touch pan and zoom to allow model exploration, on the other hand, HoloLens used its full capability of two hand manipulation to move, rotate and scale the entire model of building as per the user's choice.

5.3 Floor Visualization

Based on device type, different visualization is implemented to best suite user experience. For iOS, the screen is the canvas and for HoloLens, the user's world space should be considered and used utilized efficiently. The iOS application supports three different type of visualizations. For navigation purpose, we further introduce AR based immersive view.

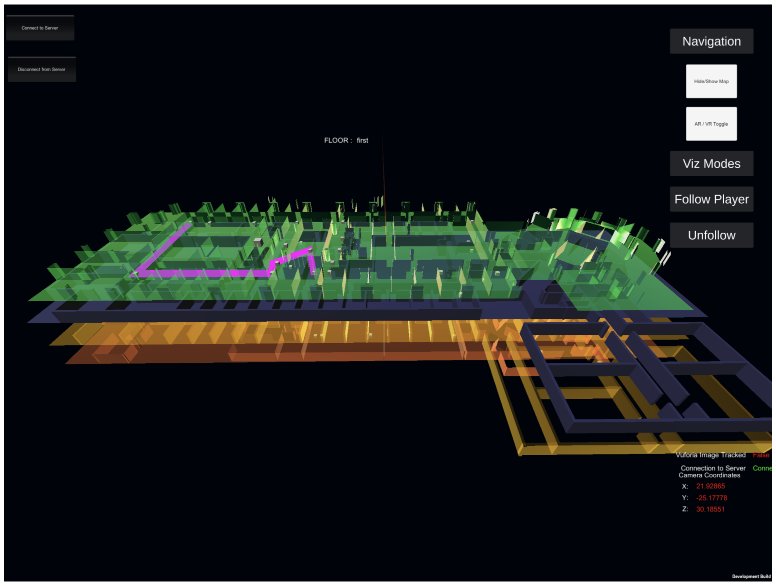


Figure 9: Isometric Visualization on iPad

5.3.1 Isometeric Visualization

Isometric visualization [2] represents how the building is made available to the server and to the other clients. Though it might not have any effect on end user, all the repositioning algorithm works based on this initial model. The coordinates of other players are received on this default view and then converted into which view the user is currently looking. The shader material used for floor model have high transparency to allow user to see through floors even from an inclined view angle.

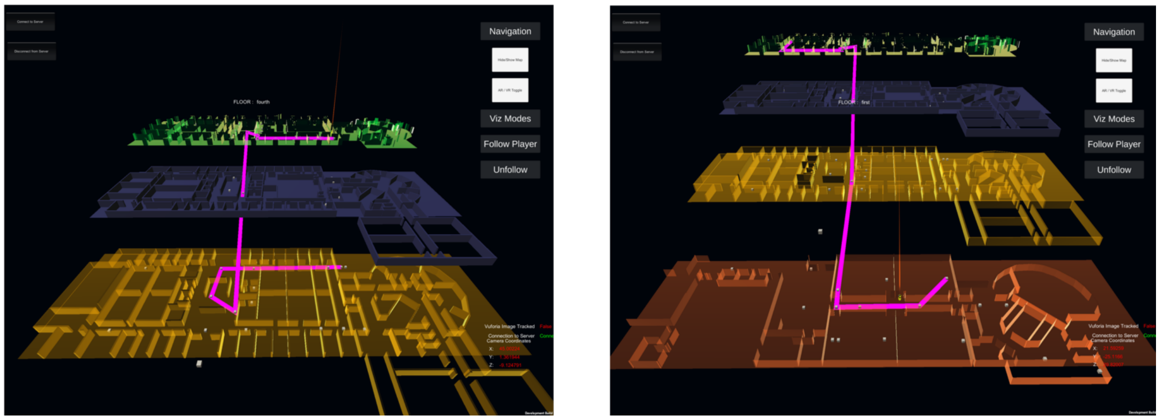


Figure 10: Staircase Visualization

5.3.2 Staircase Visualization

Staircase Visualization allows user to have benefit of 3D understanding of building like in isometric view but also allows user to cover all areas without any obstruction in between. And because it is layered as staircase, it becomes easy to recognize the floor.

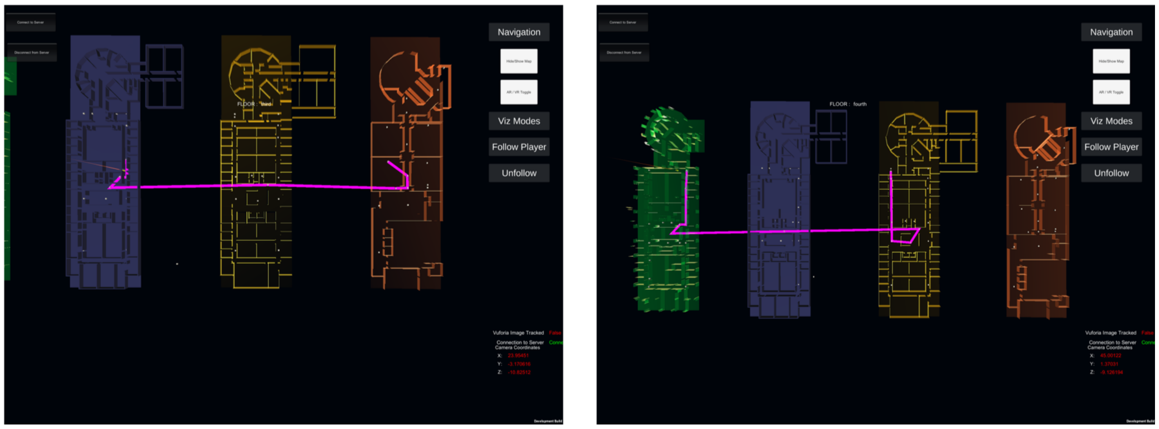


Figure 11: Orthogonal visualization on iPad

5.3.3 Orthogonal Visualization

Orthogonal Visualization [26] is basically the orthographic view of the model. This view helps understand the user's movement along the walls with greater precision. However, the height information is not revealed here. If there are certain areas with elevation, it cannot be properly understood in this view.

All the three visualization support the touch controls to zoom and pan across the map.

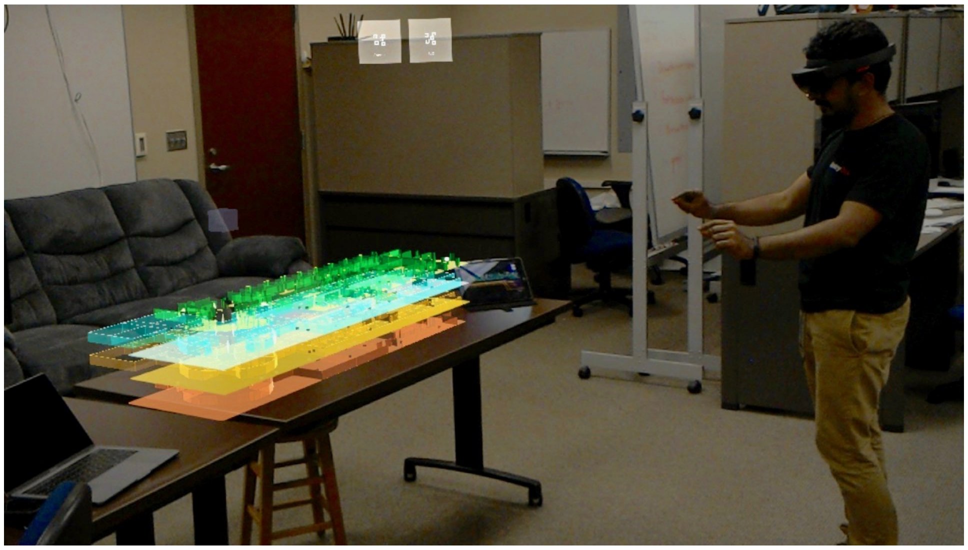


Figure 12: Two Hand Manipulation on HoloLens

For HoloLens, the 3D building model is handled differently. Just like for buttons, the maps were also assigned components to face the user at all times. No matter where the user goes, the map always followed the user and anchored itself to specific place in front of user. The $TagAlong$ component made sure that the transition is more natural. Because the user would wish to walk around the building model, to have different perpsective, speech commands are implemented to make the building stay at one place or to follow the user. "Stay" command comes handy when user wants to explore the building by not actually moving around the building whereas the follow command helps user when walking inside

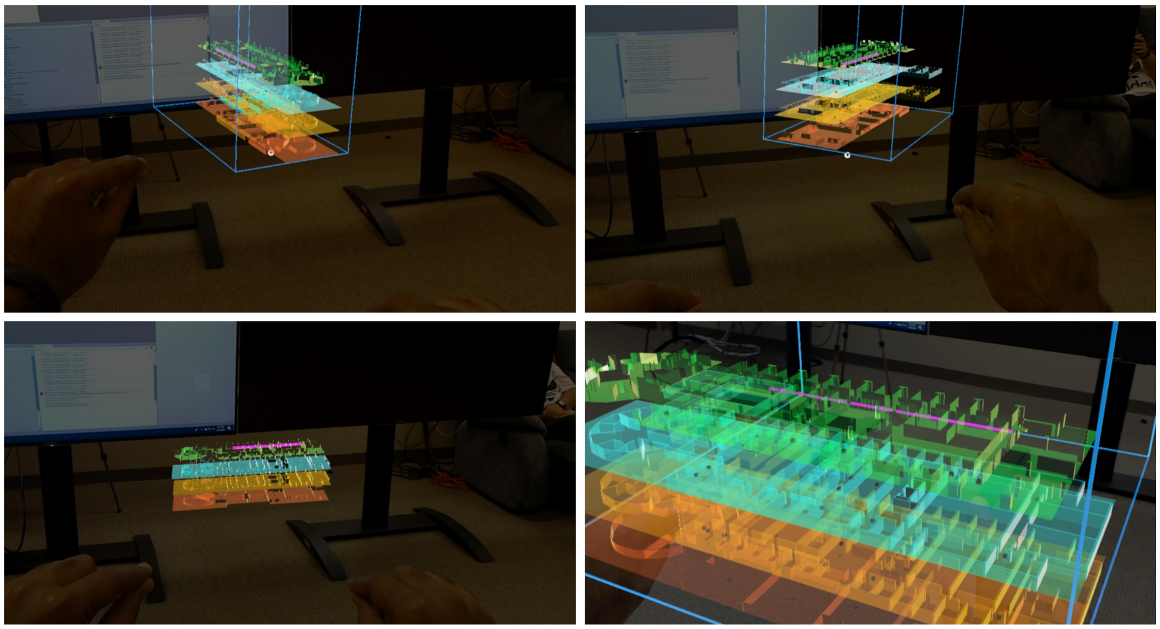


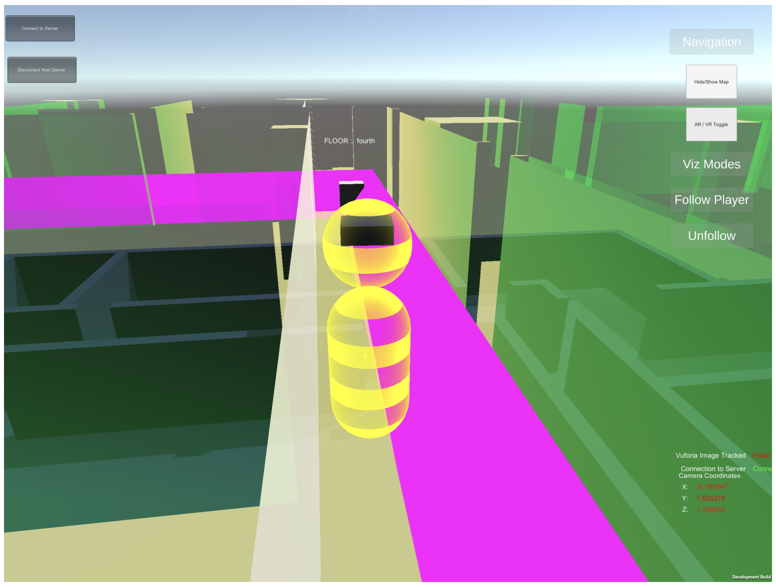
Figure 13: Rotation and Scale using Two Hand Manipulation

the building by staying around the user. Stay command also allows user to zoom the model.

Paths in both the device are drawn using the line render component. However, one difference is the line renderer is redrawn in HoloLens. Because the user moves, the 3D building inside HoloLens is continuously changing its location based on user's position. Line Renderer does not stay intact with its parent component in Unity and hence appears drifitng away from the model as user moves away. To avoid this, the line renderer on HoloLens is redrawn on each update. The path on HoloLens and iOS device is shown by drawing the path through each node inside the selected pathway.

5.3.4 AR Immersive View

As iOS users dont have 3D manipulation of model as in HoloLens, we added AR based immersive view for mobile devices. It uses device sensor data to control camera angles and gives third person view to user while navigating inside the building. This view helps user to look around the building and also get precise turn by turn guidance. The images shown



Vuforia Image Tracked False
Connection to Server Conne
Camera Coordinates
X: -6.191647
Y: 1.624219
Z: 1.336615

Figure 14: AR Immersive Visualization on iPad Pro

below demonstrates how well the immersive view blends with real world scenario.

Along with different visualizations, we have added a feature of following a player. This feature allow users to follow another user by selecting a dedicated button on screen. This feature draws shortest path to another user and keeps updating in real time as the users gets near or away from each other.

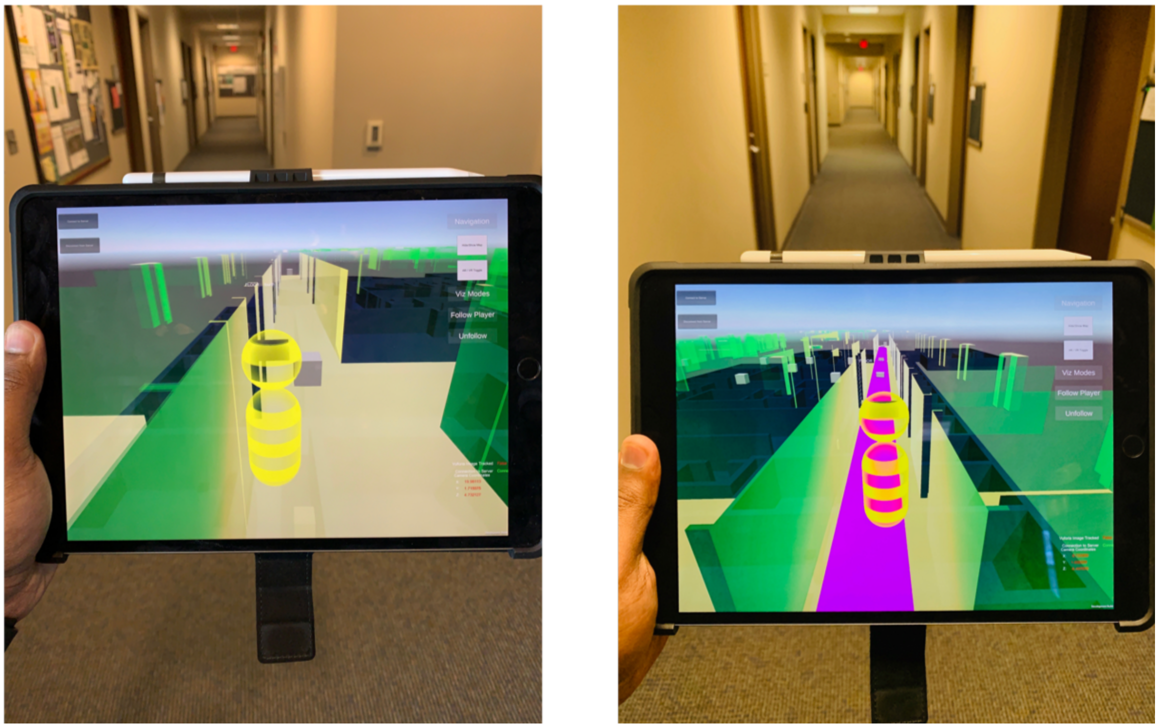


Figure 15: Real world AR Immersive View on iPad Pro

CHAPTER 6: EVALUATION

6.1 User Study

We conducted a user study with four different tasks to understand several aspects of this project. The concerns include effect on tracking if device is handled differently, user's tendency of using such application, accuracy in real life scenario, etc. User suggestions were also taken.

Five participants were involved in this user study. All of them were students of ages from 24 to 28. They were called in the Visualization Lab in Woodward Building. Two of them had visited this building before. Nobody had experience using HoloLens before and were given HoloLens to learn basic operations before starting the user study.

6.2 Procedure

To understand overall effectiveness of the system, there were total four tasks involved exploring different functionalities.

Task 1: AR Immersive Nav System. The users were asked to go till Room 435F. Users were asked to switch between immersive view and default view.

Task 2: Viz Exploration. The users were asked to go till Room 130 situated in first floor from our visualization lab on fourth floor via stairs. While navigation, users were asked to use any of the three visualization they feel comfortable with.

Task 3: Friend Finder. Group of two users were considered in a task. Both the users registered the devices and then went to random location inside and he building. The task was to find each other out using the application on iPhone or iPad. Users were again asked to switch between any three visualization.

Task 4: Rescue Coordination. One user was asked to take iPhone/iPad and roam around on third floor of the building. Another user acting as a rescue coordinator was given a HoloLens and they both could talk to each other using cellular call. Rescue coordinator was assigned a task to assist the user to reach a certain location on the map on fourth floor. They used cellular call for communication. Rescue Coordinator was free to use model navigation commands. Also he could scale rotate and reposition the map for his convenience.

Tasks 1 and 2 were done individually. The participants were given either iPhone or iPad. The devices were registered with image target and were connected to server. The navigation system was set. Server kept track of all position and rotation of users throughout the session.

Tasks 3 and 4 required two people to be together. To make task less exhaustive, the participants were asked to roam on one floor only. For task 4, HoloLens and iPhone were handed to both user respectively and they were asked to talk through the phone while using the maps. The rescue coordinator was in the lab in 4th floor looking at the trapped user's location on third floor. Rescue coordinator was asked to use the two hand manipulation to scale and guide the user to reach a specific point on 4th floor.

6.3 Expert Feedback

To get a psychological aspect of our project, we gave a demo to Dr. Alexia Galati, a professor from Department of Psychological Science. She performed similar tasks but three times to understand how movement of a person can differ the tracking and also with all the three devices - iPhone Xs Max, iPad Pro and HoloLens. Her review states, “This is an innovative project with the potential to facilitate collaborative spatial tasks (e.g., navigation, search-and-rescue).”

We also received valuable feedback on improvements and areas to focus for future iterations. During the demo, due to some unknown reasons, the voice commands were not getting accurately accepted. At this time, restarting the application resolved the issue. Also there was some drift in location which was getting accumulated over time, which needs to be improved by introducing more image targets in the buildings to re-caliberate user’s location. She also recommended different players to have different colour schemes which will make it easy to distinguish which users to track. Professor also noticed that minimizing the app affects the system tremendously and loses the position information. This use case should be handled properly. She also mentioned that heading of the player needs to be visualized properly which will give the user an overall idea of how the building is laid out.

6.4 Summary of Results

Different tasks gave us some useful insights. It also helped us understand which visualization users prefer to use more.

The summary of task 1 indicates that users found the implementation to be interesting. Most of them agreed that it could be used in malls to find a shop but were skeptical if this

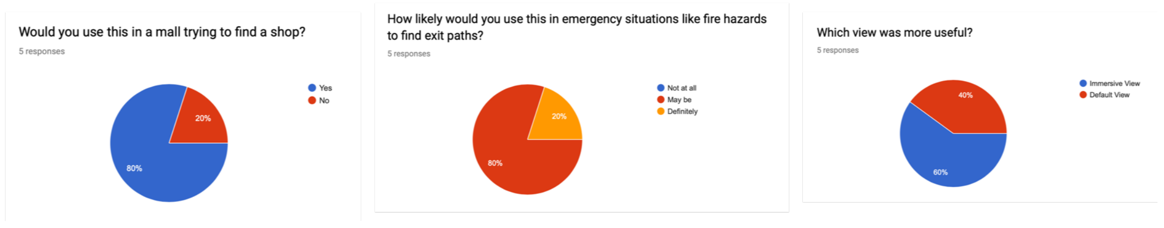


Figure 16: Summary for Task - Immersive Navigation System

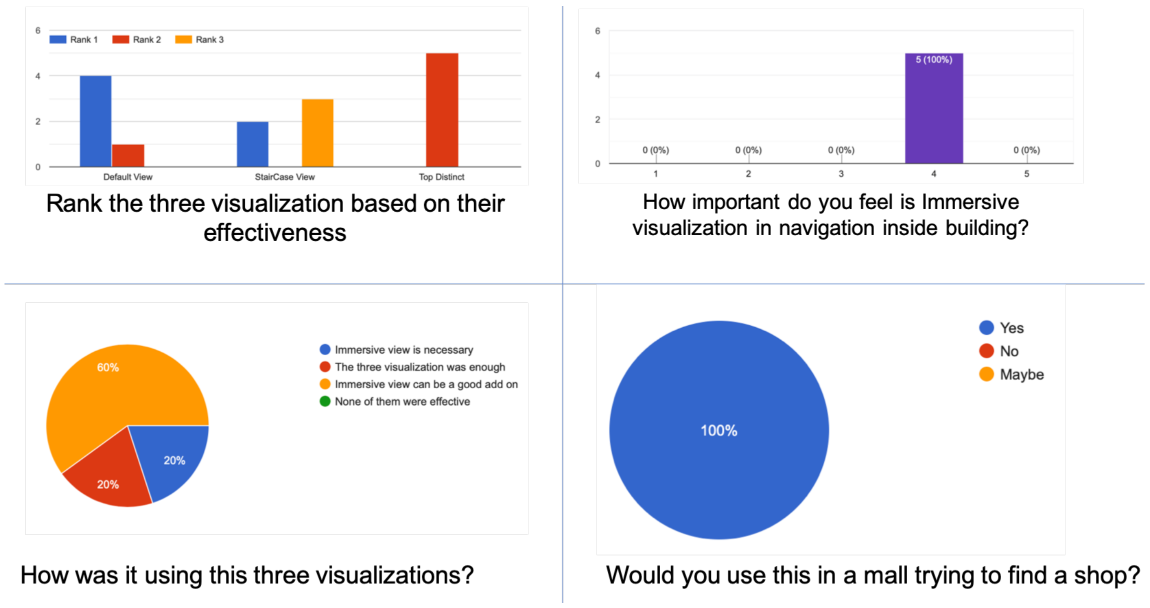


Figure 17: Summary for Task - Viz Exploration

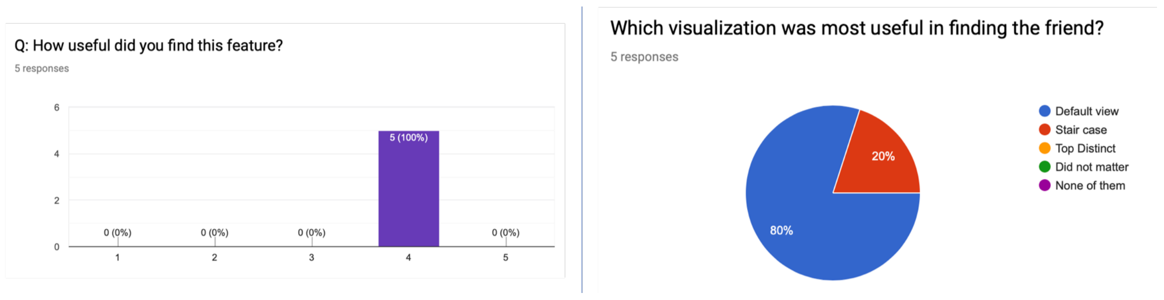


Figure 18: Summary for Task - Friend Finder

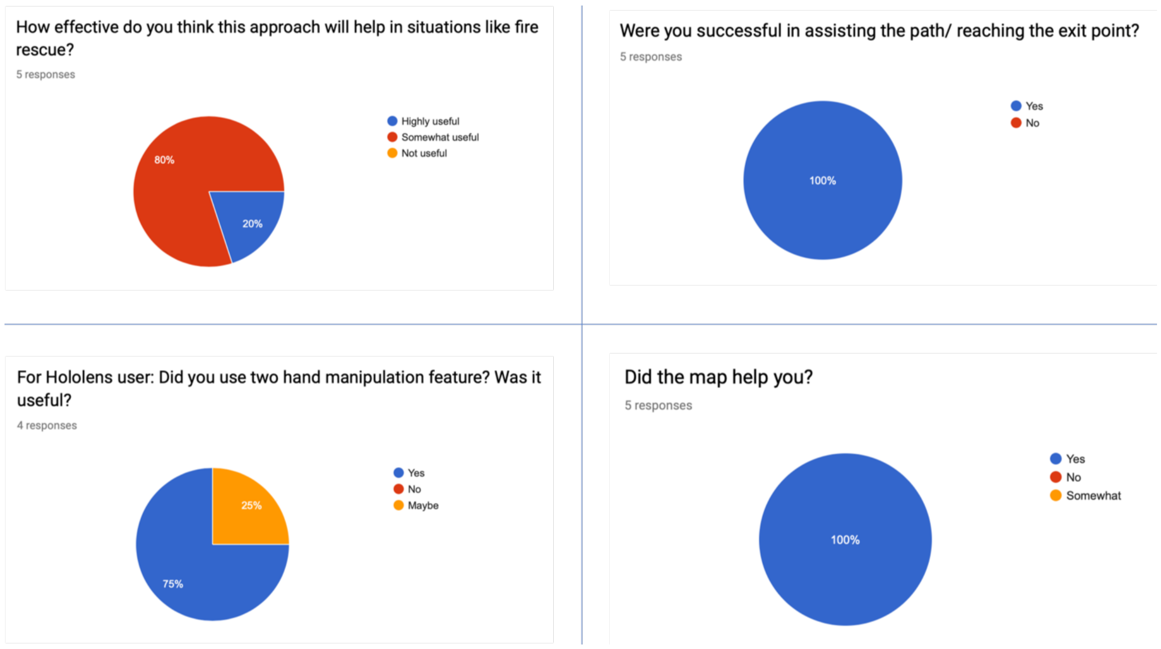


Figure 19: Summary for Task - Rescue Coordination

could be used in an emergency situation. And out of the two views, majority of them liked the immersive view.

For task 2, because the users were asked to three floors down, all of them agreed that this system with different visualization can be helpful in navigating inside the mall. As for this task, we restricted them not to use the immersive view so that we could get a fair idea if three visualization can be useful. However, the feedback collected show most of the users prefer having immersive time but could still be fine without having it.

For the task 3, all the participants agreed the technique to be extremely useful. On asking which location or situations can it be useful, we got responses like - “Emergency Services like 911 to track down victim”, “Finding items in big malls like Walmart or Lowe’s or Costco”, “Malls, Offices, parking areas” and “Shopping malls, Offices, multilevel parking lots.” Out of the three visualization, they preferred using default view. But this is because all the participants were mostly finding the friend on the same floor.

For Rescue Coordination task, participants agreed the map was useful and they were able to accomplish the task. For HoloLens user, the two hand manipulation really helped as it could allow the user to zoom and understand other user's perspective and guide them accordingly. When asked how effective the this appriach will work for emergency situation, surprisingly, most of them said "somewhat useful". When asked for feedback and suggestion, we got responses like - "Does the users last location be locked in case of device damaged in emergency situation?" and "How would you overcome visibility in case of smoke in fire situation?"

CHAPTER 7: CONCLUSION AND FUTURE WORK

The developed cross-platform framework uses Unity and its UNet High level API (HLAPI) to build a plug and play framework to help quick development of applications focussing on shared augmented reality experiences. With custom building navigation system, the application determines best possible path from user location to selected destination. Application also stores different possible paths for collaborative tasks in future implementation. The project, by using devices of different platform, an iOS device and Microsoft HoloLens and integrating networking behaviour, shows it is pretty easy to develop a networked cross platform project. Use of Vuforia makes the project portable to Android and other platforms which support Augmented Reality supporting Vuforia.

The project have been tested with three HoloLens, two iPad Pros and one iPhone Xs Max and had been running smoothly. The project along with one server can currently handle requests of 10 concurrent users. UNet networking, after Unity 5.6 release increased the concurrency upto 128 connections thus allowing the project to be more extendable on code optimization.

Benefits of Augmenting Reality cannot be unseen. Not only the technology is new and surprising to the first time users, but real use cases of augmented reality are branching endlessly. From retail, entertainment, military, healthcare, disability accessibility etc, there are hardly any field where AR cannot be helpful. Growing libraries and low hardware cost enables developers to explore the field easily. From personal experience, it had been great to learn Unity and relate it with iOS development which I had been familiar with. Not only AR enables visually appealing applications but also encourage users to collaborate other fields of science with AR. For example, we can use object detection, machine learning libraries to add more capabilities to the applications as needed.

This project also emphasizes on utilization of same data by different types of devices. Based on device type, the type of visualizing the data may change drastically in AR-enabled devices. While HoloLens requires map displayed in front of user the whole time, the same app on an iPhone doesn't require the application to render the camera feed even though the camera was being accessed for image target detection and for the device to understand the surface and the motion of the device. This is a crucial part for developing apps which are cross-platform. World see through need not be explicitly included unless required.

In future, apart from improvements on feedback, I would like to switch the networking component to use more generic www request with a separate server hosting APIs or web sockets so that the project will be decoupled with Unet Server protocols. This will allow Project developed on Unity to sync with a project separately developed on Xcode or Android Studio as the devices will only have to deal with json location data of other clients. Use of better textured models can help the app look realistic and more intuitive to users.

REFERENCES

[1] B. Bach, R. Sicat, J. Beyer, M. Cordeil, and H. Pfister. The hologram in my hand: How effective is interactive exploration of 3d visualizations in immersive tangible augmented reality? *IEEE Transactions on Visualization and Computer Graphics*, 24(1):457–467, Jan 2018.

[2] D. Ben-Haim, G. Lappan, and R. T. Houang. Visualizing rectangular solids made of small cubes: Analyzing and effecting students' performance. *Educational Studies in Mathematics*, 16(4):389–409, Nov 1985.

[3] H. Blunck, M. Kjrgaard, T. Godsk, T. Toftkjr, D. Lund, and K. Grnbk. Empirical analysis and characterization of indoor gps signal fading and multipath conditions. *International Technical Meeting of the Satellite Division of the Institute of Navigation (GNSS)*, 01 2009.

[4] S. Butscher, S. Hubenschmid, J. Müller, J. Fuchs, and H. Reiterer. Clusters, trends, and outliers: How immersive technologies can facilitate the collaborative analysis of multidimensional data. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 90:1–90:12, New York, NY, USA, 2018. ACM.

[5] E. Cay, Y. Mert, A. Bahcetepe, B. K. Akyazi, and A. S. Ogrenci. Beacons for indoor positioning. In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–5, Aug 2017.

[6] T. Chandler, M. Cordeil, T. Czauderna, T. Dwyer, J. Glowacki, C. Goncu, M. Klapperstueck, K. Klein, K. Marriott, F. Schreiber, and E. Wilson. Immersive analytics. In *Big Data Visual Analytics (BDVA), 2015*, pages 1–8, 2015.

[7] H. Chen, A. S. Lee, M. Swift, and J. C. Tang. 3d collaboration method over hololens^{TM} and skype^{TM} end points. In *Proceedings of the 3rd International Workshop on Immersive Media Experiences*, pages 27–30. ACM, 2015.

[8] M. Cordeil, A. Cunningham, T. Dwyer, B. H. Thomas, and K. Marriott. Imaxes: Immersive axes as embodied affordances for interactive multivariate data visualisation. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, UIST '17, pages 71–83, New York, NY, USA, 2017. ACM.

[9] M. Cordeil, T. Dwyer, K. Klein, B. Laha, K. Marriot, and B. H. Thomas. Immersive collaborative analysis of network connectivity: Cave-style or head-mounted display? *IEEE Transactions on Visualization and Computer Graphics*, PP(99):1–1, 2016.

[10] Z. Cui, S. Sen, S. K. Badam, and N. Elmqvist. Vishive: Supporting web-based visualization through ad-hoc computational clusters of mobile devices. *Information Visualization*, 2018.

[11] D. Datcu, M. Cidota, H. Lukosch, and S. Lukosch. On the usability of augmented reality for information exchange in teams from the security domain. In *Intelligence and Security Informatics Conference (JISIC), 2014 IEEE Joint*, pages 160–167, Sept 2014.

[12] N. Elmqvist and P. Irani. Ubiquitous analytics: Interacting with big data anywhere, anytime. *IEEE Computer*, 46(4):86–89, 2013.

[13] T. Horak, S. K. Badam, N. Elmqvist, and R. Dachselt. When david meets goliath: Combining smartwatches with a large vertical display for visual data exploration. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, 2018.

[14] U. Kister, K. Klamka, C. Tominski, and R. Dachselt. Grasp: Combining spatially-aware mobile devices and a display wall for graph visualization and interaction. *Comput. Graph. Forum*, 36(3):503–514, June 2017.

[15] O. H. Kwon, C. Muelder, K. Lee, and K. L. Ma. A study of layout, rendering, and interaction methods for immersive graph visualization. *IEEE Transactions on Visualization and Computer Graphics*, 22(7):1802–1815, 2016.

[16] B. Lee, P. Isenberg, N. H. Riche, and S. Carpendale. Beyond mouse and keyboard: Expanding design considerations for information visualization interactions. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2689–2698, Dec. 2012.

[17] S. Lukosch, H. Lukosch, D. Datcu, and M. Cidota. On the spot information in augmented reality for teams in the security domain. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, pages 983–988. ACM, 2015.

[18] E. Mahfoud, K. Wegba, Y. Li, H. Han, and A. Lu. Immersive visualization for abnormal detection in heterogeneous data for on-site decision making. In *HICSS*, 2018.

[19] T. Mahmood, E. Butler, N. Davis, J. Huang, and A. Lu. Building multiple coordinated spaces for effective immersive analytics through distributed cognition. In *2018 International Symposium on Big Data Visual and Immersive Analytics (BDVA)*, pages 1–11, 2018.

[20] S. Nilsson, B. Johansson, and A. Jonsson. Using ar to support cross-organisational collaboration in dynamic tasks. In *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pages 3–12, Oct 2009.

[21] K. Nuaimi and H. Kamel. A survey of indoor positioning systems and algorithms. pages 185 – 190, 05 2011.

[22] R. Sicat, J. Li, J. Choi, M. Cordeil, W. Jeong, B. Bach, and H. Pfister. Dxr: A toolkit for building immersive data visualizations. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2018.

[23] D. Sims. See how they run: modeling evacuations in vr. *IEEE Computer Graphics and Applications*, 15(2):11–13, Mar 1995.

[24] M. Speicher, B. D. Hall, A. Yu, B. Zhang, H. Zhang, J. Nebeling, and M. Nebeling. Xd-ar: Challenges and opportunities in cross-device augmented reality application development. *Proc. ACM Hum.-Comput. Interact.*, 2(EICS):7:1–7:24, June 2018.

[25] W. Usher, P. Klacansky, F. Federer, P. Bremer, A. Knoll, J. Yarch, A. Angelucci, and V. Pascucci. A virtual reality visualization tool for neuron tracing. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):994–1003, Jan 2018.

[26] X. Wang, J. Lim, R. T. Collins, and A. Hanson. Automated texture extraction from multiple images to support site model refinement and visualization. 02 1970.

[27] C. Ware and P. Mitchell. Visualizing graphs in three dimensions. *ACM Trans. Appl. Percept.*, 5(1):2:1–2:15, Jan. 2008.

[28] M. Wu and V. Popescu. Efficient vr and ar navigation through multiperspective occlusion management. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2018.

[29] Y. Yang, T. Dwyer, B. Jenny, K. Marriott, M. Cordeil, and H. Chen. Origin-destination flow maps in immersive environments. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2018.

[30] Y. Yang, B. Jenny, T. Dwyer, K. Marriott, H. Chen, and M. Cordeil. Maps and Globes in Virtual Reality. *Computer Graphics Forum*, 2018.