City, University of London MSc. in Games Technology	
Project Report	
Name	Sean Feeley (adbr449)
Year	2022
Project Title	CardsAR, A Social Card Game Set In A Shared Virtual World Using Multi-Modal Smartphone Based Augmented Reality
Supervisor	Dr Chris Child

By submitting this work, I declare that this work is entirely my own except those parts duly identified and referenced in my submission. It complies with any specified word limits and the requirements and regulations detailed in the assessment instructions and any other relevant programme and module documentation. In submitting this work I acknowledge that I have read and understood the regulations and code regarding academic misconduct, including that relating to plagiarism, as specified in the Programme Handbook. I also acknowledge that this work will be subject to a variety of checks for academic misconduct.

Signed:

Sean Feeley

Abstract

This report details the design, build and evaluation process for an augmented reality (AR) iOS multiplayer card game called CardsAR. The game has been developed in the Unity game engine using Lightship ARDK, Agora Voice SDK and PubNub messaging API. Players in the game are placed in a 3D virtual room around a circular table. 3D avatars of each player are used to show where each device is orientated in the game space. Placed in the centre of the table is a deck of cards that can be used to play card games. An audio call is also started for all players so that they can communicate while playing any card game they wish. The original plan for CardsAR included a video call but this could not be completed because of its reliance on third party tools. AR on smartphones is held back by toolsets that are designed for only the most common feature sets in the marketplace.

Keywords

Augmented Reality, Unity, Game Design, Card Game, Multiplayer

Table of Contents

Chapter 1 – Introduction and Objectives

1.1 Introduction

This report covers the design, build and evaluation of an augmented reality (AR) social game in order to fulfil the Individual Project of the MSc in Computer Games Technology at City, University of London. The project was undertaken over six months from late June to early December 2022. The outcome of the project was an iOS game that could run on iPad and iPhone devices running iOS/iPadOS 15 or above. The resulting game was called CardsAR. Difficulty with the Lightship ARDK and the Agora Video SDK led to significant compromises on the original design. These compromises could have been avoided if the scope for CardsAR was reduced. This would have given time for customisations to be made to the third party tools.

The primary design goals of CardsAR were twofold: to create an AR based card game using the Unity game engine and to provide a social space more engaging than a standard video call.

The game has been designed remote play. Each player logs into the game and is placed in a virtual room that, using AR technology, is anchored to their local environment. In this virtual world they can look around and interact with virtual avatars of other players and interact with the deck of cards placed on a table in the centre of the room.

CardsAR has been designed to eschew the common tropes of AR gaming by including video calling and multiplayer design that not tied to shared physical spaces. This raised significant challenges during development. During the development of the game, compromises had to be made because of the limitations of the third-party tools that were used. It is possible for the original design of CardsAR to be achieved but it would require writing multiple libraries which would have extended development time beyond the scope of this project.

1.2 Motivation

The coronavirus pandemic and its related lockdowns in 2020 and 2021, saw a large increase in online gaming as the predominant form of socialising (gamesindustry.biz, 2020). During this time, social video calling was also heavily relied upon (Yuan, 2020). As working from home arrangements have continued post lockdown video call usage has not fallen to its pre-pandemic levels. The design of CardsAR was motivated by the lack of games that use video calling as an integrated part of their gameplay.

Playing cards in their current form have been in use for centuries (The International Playing Cards society, 2007). Their ubiquity in society makes them an ideal foundation to base this AR game around. If the appropriate set of card interactions are correctly modelled, it allows for players to decide their own games and rules. Players can also drop in and out without breaking any code-defined gameplay rules.

After the first waves of the pandemic, businesses like Meta started to invest heavily in virtual reality devices as a way for users to socialise remotely (Meta, 2021). During a company rebranding video Mark Zuckerberg explained that he saw smartphones as a part of the old

internet and virtual reality headsets as the new primary internet device of the future (Meta, 2021). VR headsets currently have a high cost of entry, both socially and financially. By comparison, far more people have smartphones that are capable of augmented reality (bankmycell.com, 2022). This project is therefore an examination of smartphone technology and its ability to provide immersive virtual reality experiences without the use of a headset.

1.3 Research Questions

The design of CardsAR aims to answer the following research questions:

1. Is it possible to build a tabletop card game using smartphone AR tools?
2. What limitations are there to user interface design when creating an AR virtual world game?
3. To what extent can smartphone AR technology be used to allow players to socially interact in a virtual world?
4. Is this method of socialising online enjoyable to casual game players?

1.4 Original Work Plan

In order to sufficiently respond to these research questions, the development phase of CardsAR was split into three builds.

Build 1 was a research build, where third party tools were investigated, and proof of concept tests were made. The code written in Build 1 was not used in the subsequent Builds 2 and 3.

Build 2 was focussed on designing the core playing card gameplay and player networking.

Build 3 was reserved for refining the look and feel of the game and integrating a video call component to the game.

A detailed list of the work plan is listed below:

1. Build 1 (June 15th – July 30th):
    1.1. Basic unity setup and devices setup for testing.
        1.1.1. iPhone 12 Mini Testing
        1.1.2. iPad Mini Testing
    1.2. Basic multiplayer session setup using Lightship ARDK
    1.3. Single hand tracking using Manomotion
        1.3.1. Skeleton Detection
        1.3.2. Gesture recognition
        1.3.3. Fine skills test
    1.4. Device pointing as interaction technique test
    1.5. Face tracking test
        1.5.1. Facial Expression Detection
        1.5.2. Face position
    1.6. Streaming video call test
2. Build 2 (August 1st – Sept 30th):
    2.1. Creation of game space, and player placement.
    2.2. Card mechanics
        2.2.1. Placement of cards on table

        2.2.2. Pick up / Drop Mechanics
        2.2.3. Add and remove cards from the players “card hand”
        2.2.4. Show single cards to individual players.
    2.3. Card Deck mechanics
        2.3.1. Shuffle and Deal
        2.3.2. Pick up multiple cards
    2.4. Player audio.
    2.5. Multiplayer session management
    2.6. Refactoring of game code.
3. Build 3 (October 1st – October 31st):
    3.1. Android Testing
    3.2. Player presence through animated avatars or floating video call screens.
    3.3. Refinement of interaction techniques based on user testing
    3.4. Improved graphical treatment using shaders, lighting and models taken from the Unity store.
    3.5. Camera parallax through face tracking.
    3.6. Player hand modelling.
    3.7. Refactoring of game code.

1.5 Work Plan Changes

Midway through Build 2, two significant issues were encountered that resulted in an alteration to the plans for Build 3. Originally Lightship ARDK was intended to be used for the multiplayer netcode, however the multiplayer sessions in ARDK are limited to players in the same physical space. Since CardsAR is designed to be played remotely an alternative design was used for multiplayer netcode. This is detailed in section 4.1.3.2.
During Build 2 it was also discovered that video calling disabled the AR camera. Therefore, the video call feature had to be reduced to an audio only call.

1.6 Beneficiaries

CardsAR is an experiment in game design that is attempting to use AR technology in new ways. The final product is a valuable proof of concept that will be useful for other AR developers in the game development community. The user experience (UX) and user interface (UI) research community will benefit from the final results as the uses of AR technology extend beyond game development.

The author is also being partially financed through this part time masters course by their employer, Territory Studio. Territory is a multidisciplinary design studio whose work ranges from creative advertising to motion graphics to post production visual effects for film and television. The studio also has a growing immersive department which has focussed on interactive exhibitions using Unreal engine thus far. In financing the author’s study, Territory is investing in the studio’s technical skillset around mobile based AR and Unity based projects.

The planned development of CardsAR incorporates 3rd party tools that are active in promoting successful use of their tools. Hand tracking toolset Manomotion promoted community projects on the YouTube channel (Manomotion, 2021). Niantic’s AR development toolset: Lightship ARDK, was released at the end of 2021. The company promotes their toolset through their

website, social media channels, community game jams and yearly awards (Niantic, 2022). If this project is successful, it could be added to each company's community showcase.

Finally, the experience of designing and utilising multiple technologies has not only broadened the authors skillset, but is also an important step forward in the author's career development as further experience with Unity and augmented reality plugins and packages allows the author to specialise in augmented reality development post-graduation.

1.7 Outcome

The outcome of this project is a build of CardsAR that can be distributed to volunteer testers via Apples Test Flight service. Test Flight allows for builds to be shared through the App store without requiring the normal authentication and evaluation checks that a complete Apple App Store submission requires (Apple, 2022).

Players have also submitted their CardsAR feedback through a Google forms questionnaire. The feedback was collected and anonymised before inclusion into the evaluation section of this report.

Chapter 2 – Context

2.1 Multiplayer Social Games

In March 2020 the UK government, announced its first full lockdown to help prevent the spread of Covid 19 (Gov.uk, 2020). Millions of people had to work from home and the only way they could spent time with friends and family was online. This created a boom in social video calling as applications like Zoom became replacements for in-person social gathering (Yuan, 2020). At the same time video game companies saw record breaking increases in demand for video game consoles as people looked for new ways to spend their free time (gamesindustry.biz, 2020). The increase in gaming was not limited to dedicated consoles, smart phone and web browser games rose in popularity at the same time (Knezovic, 2022). Covid 19 increased the national consumption of video games, but it also created a new type of video game player that did not exist before. People who do not normally play multiplayer games but were in a situation where video games were one of the only forms of group social activity.

This change in the marketplace can be observed in the rise in popularity of Among Us (Stuart, 2022). Among Us is an online multiplayer social deduction game (IGN.com, 2021)inspired by the horror movie The Thing (The Thing, 1982). In Among US players control a 2D character that has to undertake tasks along with their crewmates while trying to avoid being killed by the secret imposter crewmate who is trying to sabotage the team. Released in 2018 Among Us saw little mainstream popular attention and maintained a small audience for the next 18 months. In mid 2020 the games popularly skyrocketed and in the years since has become a multimillion dollar franchise. The games creators have attributed the games popularity to its low barrier, simple design and broad availability on multiple devices (Noclip, 2022).

As well as playing online video games, it was common for people to try to translate in person games to remote friendly format. This could be using websites to play digital versions of board games, or using video calls to organise quizzes. In many of these situations, the video calls may be continued through a rotation of digital board games (Law, 2020).

Another company that took advantage of the change in the market was Houseparty. This social network application was available on PC, Mac, Android and iOS and let users play built in games like Heads Up, Trivia and Uno while on a video call. When playing games, the video call feeds from other players would be resized on screen to make space for a gameplay window. It was started in 2016 but saw a large increase in attention during the pandemic. Houseparty was however discontinued by its parent company Epic Games in 2021 due to low player numbers (Griffin, 2021).

2.2 Card Playing Games

Playing Cards have been represented in video game since the earliest graphical user interfaces were available. In 1990 with the release of Windows 3.0, Microsoft included Windows Solitaire as a free pre-installed game (Warren, 2020). It was designed partially as way to "to soothe people intimidated by the operating system," (Garreau, 1994). The game has remained a part of the Windows since its initial release over 30 years ago and in 2019 The Strong National Museum of

Play inducted Microsoft Solitaire to its World Video Game Hall of Fame (The Strong Museum of Play, 2019)

The increase of home internet usage in the 1990s and early 2000s also saw in increase in online poker playing, growing to a $2.4 billion industry in 2005 (Newsweek, 2005). Online gambling also saw an increase in users during the Covid19 pandemic (Gambling Commission, 2021). Online poker interfaces are often web browser based and are designed around flat 2D representations of cards and poker chips.

A significant influence for the design of CardsAR is Tabletop Simulator (Tabletop Simulator, 2022). This is a PC and Mac game that allows players to play and create tabletop games in a multiplayer physics sandbox. The game is designed to allow any tabletop game to be played from within it. When installed it comes with royalty free games like chess, blackjack and mah-jong, but it has an open design that allows for players to easily develop their own games with imported 3d models and custom scripting. Table Top simulator also comes with a standard interface for playing cards. Compared to browser based online poker, Tabletop simulator has a high barrier to entry. At the time of writing it costs £14.99 at full price. In order to handle the physics simulations necessary for gameplay features like realistic dice rolls the game also has non-trivial hardware requirements (System Requirtements Lab, 2022).

During development a new game called All On Board was successfully crowdfunded on Kickstarter. All On Board is a VR game similar to Table Top simulator, that designed to be platform for users to create their own board games in virtual world (The Game Kitchen, 2022).

The market for playing cards games skews either to simple web browser interfaces or high end realistically simulated tabletop games. There is currently no successful 3D environment for playing cards that can be played on smartphones.

2.3 Augmented Reality

Augmented reality can be defined as an interactive experience that incorporates real word and digital content. Since the term was coined in 1990 (Interactive Design Foundation, 2020) companies have experimented with different use cases with different hardware input devices. In 2006 the Wearable Computer Lab at the University of South Australia created AR Quake. The game used a custom build backpack and headset to allow players to play a version of popular first-person shooter Quake projected into the real world through a head mounted display (Wearable Computer Lab., 2006).

It wasn't until the release of the first iPhone 2007 that the public would have access to an internet enabled device with a camera that could be used in AR tools and games. For the first few years, AR was used in novelty applications that weren't integrated into the daily lives of users (Sung, 2012). That changed with the release of Pokémon GO in 2016, one of the most popular and profitable augmented reality app of all time (Perez, 2016). Pokémon GO's core gameplay loop consists of players travelling to real world locations to then use their catch virtual pocket monsters that appear in the real world as seen through the phone camera (Chamary, 2018).

2.3.1 AR Survey

When researching the viability of CardsAR, a survey was taken of every popular augmented reality app on iPhone and Android. Apps were identified by searching both the google play and apple app store. Articles were also collected from the past 5 years that showcase AR apps.

The methods used to collect AR apps in the survey have limitations. Only apps that were currently available to download on either the Play store or App store were considered. This means that any app released in the last decade that has been discontinued were not reviewed. Reviewing the features of a discontinued app would be too unreliable to include in the survey. This leaves the possibility that an app with similar functionality to CardsAR may have already existed but has been removed from sale. Also, by limiting the survey to published apps, research projects and unreleased prototypes from environments like game jams, have not been covered.

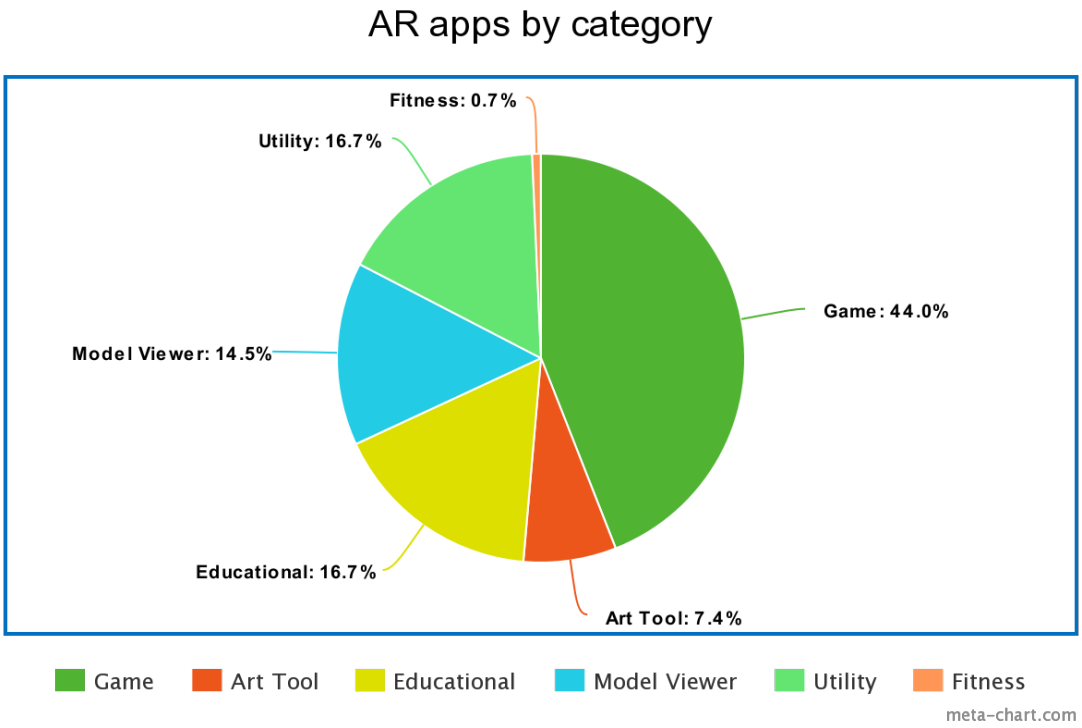
In total 149 smartphone AR apps were collected in the survey. Each app was grouped by app category and movement style.

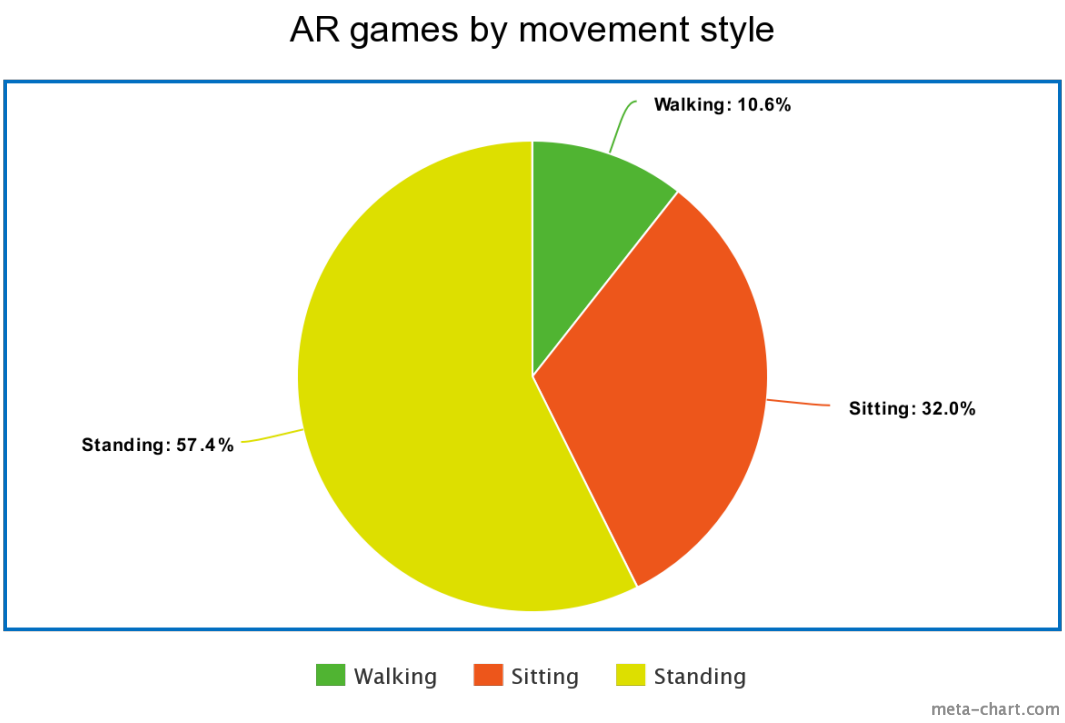
The AR app categories created for this survey are Game, Art-Tool, Educational, Model Viewer, Utility and Fitness. These size categories were based on the results of the survey and were a way to meaningfully differentiate the results.

Category	Description
Game	Designed primarily as a game
Art-Tool	Designed to be a self-expression tool. Examples include drawing apps and face filter apps.
Educational	These apps are designed to tell a story or convey information through 3D models.
Model-Viewer	These apps let users place 3D models in their environment. This can be with a list of predefined novelty models like dinosaurs or vintage cars. Or it can be a community app that lets you look at models other users have created.
Utility	These apps achieve a task that AR is uniquely suited to. For example taking the measurements of a room.
Fitness	Fitness apps are primary for the tracking or fitness activities or for guidance on how to perform certain exercises.

Each game was also grouped by another criteria, movement style. The categories are seated, standing, and walking. Seated apps can be used while seated and do not require the user to stand or move significantly while using. Standing apps require or at least encourage players to move around a space to either look at multiple focal points on the space or look at a focal point from different perspective. Finally walking apps require that the user move from location to location. Walking apps normally incorporate GPS into their systems.

2.3.2 AR Survey Results





From these results it can be seen that “standing” is the most common movement style for AR games. Standing games try to take advantage of the 3d space that a person is currently placed. A typical example of such a game is Euclidean Lands (kunabi brother, 2017) or Arise (Climax

Studios, 2017), both are puzzle games that have the player walk around a 3D model that they project into the middle of a space. To solve puzzles players must look at the model from the different positions and angles to enable gameplay features.

CardsAR can be considered a sitting game. A common gameplay feature of sitting games is the projection of a 3D game board onto a flat surface in front of the player. Similarly to CardsAR games like Jenga AR (Free Range, 2022) and Lego Hidden Side (LEGO Systems, 2022) attempt to create a digital simulation of a real like tabletop game. Many of the planned features for CardsAR were not identified in the survey. No game used playing cards as its focus, and none of the games focussed on socialising as a primary feature. Every game also displayed the 3D models overlaid onto the real world, none of the games took place in a virtual world like the one planned for CardsAR.

2.4 Hand Tracking

Hand tracking from the world facing camera has been considered as an interaction method. Some models of smartphone contain LIDAR scanners which could be used to accurately tracks hands and fingers (Apple, 2020). This feature is only available on new high-end iPhones Pro and iPad Pro models, which would negatively impact the availability of users to test this feature. Moreover, as CardsAR has been designed as a low-cost more broadly available alternative to high end VR devices, it should be designed with features available to the most number of devices.

Third party tools do exist for hand tracking from a single source RGB image. Manomotion was selected as the platform to build this feature upon. This choice was made as the company has a Unity toolset for hand skeleton from the front facing camera (Rutegard, 2021).

2.5 Face Tracking

Face tracking is the most common AR use for front facing camera of a smartphone. Across the AR survey, this was the only use for the front facing camera. Glasses retailer Warby Parker has an app that lets users try on glassed in real time with a projected 3d model (Warby Parker, 2022). Social media apps Snapchat, Instagram and TikTok all allow users to take pictures and record videos of themselves with 3d models and effects projects onto their tracked faces (Facebook, 2017) (Snapchat, 2022) (TikTok, 2022).

Chapter 3 – Methods

3.1 Development Environment

3.1.1 Unity

Unity has been chosen as the game engine to develop CardsAR within. This was selected for its extensive support for smartphone-based AR development. Unity has multiple options for AR plugins as well as an active community and marketplace for models and plugins. This choice aligns with choices made by game development studios, both the Niantic and Rovio Entertainment develop their AR games in Unity (FRANCIS, 2016) (DAVENPORT, 2022).

3.1.2 AR Toolset

Multiple software development kits (SDK) are available in Unity for AR. Unity has support for both iOS's ARKit and Android's AR core. CardsAR was developed primarily on an iPhone, but it be tested on Android during Build 3. Therefore a cross-platform AR SDK solution is required. AR Foundation meets the requirements, it allows for AR apps for Android and iOS to be built from the same code. This SDK however doesn't have the breadth of features that Lightship ARDK has.

Lightship ARDK from Pokémon Go creators Niantic is a cross platform AR toolkit for unity that, in adds to the feature sets found in ARKit, AR Core, and AR Foundation (Niantic, 2022). In addition to 3D plane detection, Lightship provides built in multiplayer netcode and session management. The toolkit also provides an in-editor world simulator for Unity. This allows for AR features to be tested without needing to build to an external device. Both these features allow for more time to be spent on feature development and less time on multiplayer testing.

3.1.3 Video and Audio

For the video call component for CardsAR, Agora was chosen. The Agora Video and Voice SDKs are both available as plugins in the Unity Asset store for free. This is the only video call plugin in the Unity store that has been marked as a "Verified Solution" by Unity. Verified Solutions are Unity packages that have been are compatible with the latest versions of Unity, and have been recognised as having long term dependability (Agora.io, 2022).

3.1.4 Hand Recognition

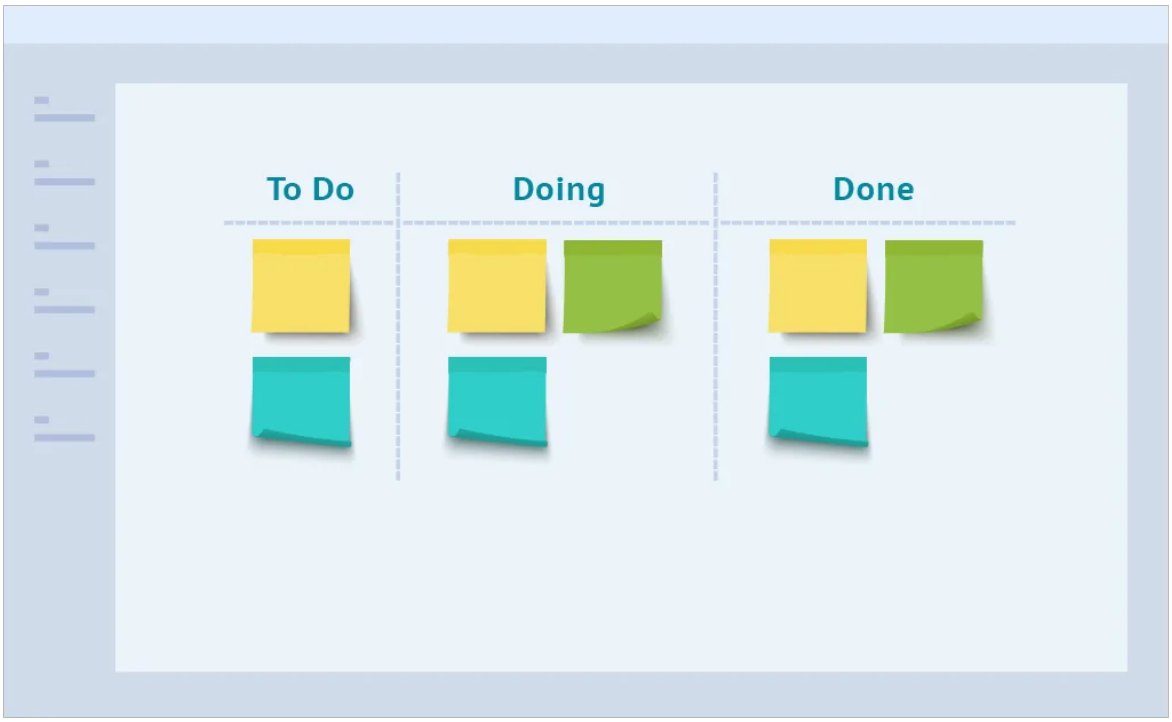
For hand gesture recognition the ManoMotion hand tracking SDK was used. This was selected as it was free to use in a university project, and it provided enough demo projects that the feasibility of hand gesture based control could be tested without significant work (ManoMotion, 2022).

3.2 Software Development Practices

Throughout development a Git repo has been maintained on GitHub.com. Build 1 is focussed on research and third-party plugin experimentation which was tracked in the CardsAR repository as no code from the tests was be carried through to Builds 2 and 3. Work on Cards AR started being

tracked with the feature development in Build 2 and 3. At the end of every day a commit was made to the main repository and uploaded to Github.com

All builds have been planned with the software project management tool Jira. The three builds and their associated features have been entered into the Jira database. Two views were set up for managing the development of CardsAR, a Kanban board and a roadmap. Kanban is a framework for managing Agile software development projects within a team. Tasks are displayed on a board with columns for their status (“to do”, “in progress”, “done”), and as the project progresses tasks and moved between columns (Digite, 2022).



The roadmap view allows for planning of the project timeline. Each feature has an allotted time, and each week the allocated times are adjusted depending on the work achieved that week.

[Roadmap screenshot]

3.3 Design

3.3.1 Scope

CardsAR is intended to be as close an analogue to real world card playing as possible. The game contains a single deck of cards. As no predefined game rules have been designed into CardsAR the interaction model attempts to cover all actions a player may need. Taking inspiration from video call social gaming, CardsAR has also be designed to be easy to drop in and out of. Players

should be able to join and leave a game without breaking the current ongoing game. To keep development time down, there is no multiplayer session interface, the final build auto-joins players into the gameplay session with hardcoded authentication credentials.

There is no 3D physics model. Cards are either held by a player, or be placed on the table. A dedicated physics model is unnecessary for the cards AR feature set. Also cards do not animate between positions, they are either statically placed or dragged by players across the table.

3.3.2 Card Gameplay Model

The card gameplay model has the following features.

- Cards can be stacked on top of each other or individually laid out.
- Stacks of cards can be fanned across the table to allow users to see each card's suit and number.
- Cards can be flipped individually or a whole stack can be flipped.
- Cards can be visible to all players or just a single player.
- Cards and stacks card can be highlighted by players
- When one player highlights a card/stack, no other player can highlight it.
- When a card/stack is highlighted one or more cards can be picked up and held by the player.
- A held card/stack can be placed onto the table again or added to an existing stack.
- Stacks of cards can be sorted and shuffled.

This feature set allows for cards to be arranged and placed in any order on the table. Card visibility allows users to have a card "hand", a set of cards only visible to themselves. A dedicated deal card function was not be added but card dealing can be achieved through picking up a stack of cards from a larger stack.

3.3.3 Environment

When players start CardsAR they set up their AR environment. As is common in most AR games, the virtual world needs to be aligned with the real world. Most AR games start with the user looking around their environment to allow the AR system to detect surfaces. The player is then asked to player a virtual marker on a detected surface. This marker is used to orientate the virtual objects or gameplay area.

CardsAR starts off with a virtual marker placement. This marker is be used to orientate the virtual table on which the cards are placed.

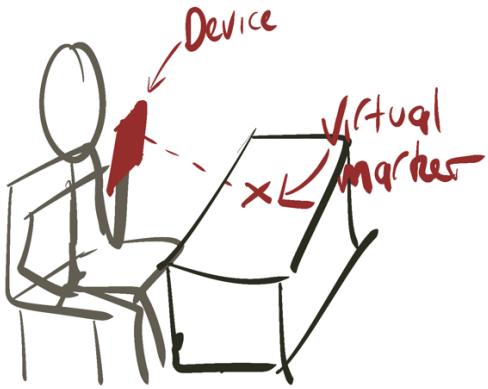


Figure a: Player places virtual marker in real world

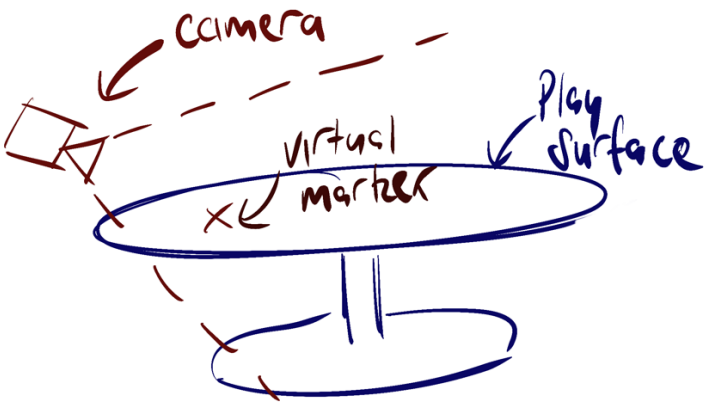


Figure b: Virtual marker sets position of table

The real world camera feed is also used by the AR tools to place the table. Contrary to most AR applications, the camera feed is not shown during gameplay after the marker has been placed. The entire game takes place in a virtual room that players join. Setting the game in a purely virtual room allows for other players to be represented in the game without needing to consider each other's physical spaces. If the camera feed was used as the gameplay backdrop, moving player avatars would break the AR illusion as their movements wouldn't be restricted by each other's physical space. If he physical environment was used to display cards, players would be required play on tables of a specific size, a virtual room is a simpler solution for inconsistencies between player environments.

3.3.4 Interaction Models

Built into the development process of CardsAR is an investigation into different interaction techniques. The first investigation was into hand recognition and whether players can hold their phone with one hand and interact with cards using pinch/grab gestures with their free hand. This was not feasible, so the primary interaction technique is a combination of on-screen buttons and using the phone are a pointing device for selecting cards.

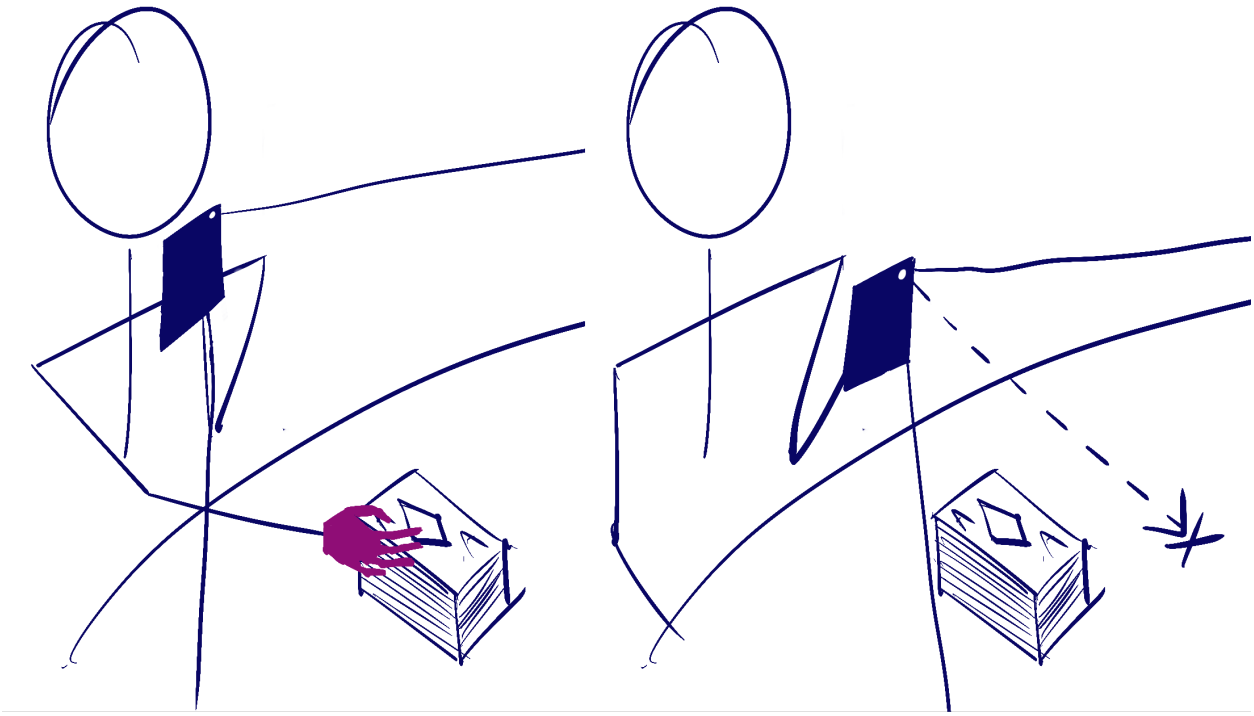


Figure c: Using the camera to detect the other player hand (left) Using the camera to point to items in on the play area (right)

Chapter 4 – Results

4.1 Development

4.1.1 Build 1

4.1.1.1 Gesture Recognition

During the research stage of this project, no examples of hand tracking as an AR gameplay method were found. Therefore, this was the first code tested when doing the first build . Manomotion had a fully featured Unity demo that could be built and run on the primary testing device, an iPhone 12 mini (ManoMotion, 2021).

The build was successful, pinching and grabbing in 3D space were reliably detected, however it did not work ergonomically work with the planned gameplay for CardsAR. The testing iPhone is equipped with two camera sensors, a standard and an ultra-wide. When using the hand tracking demo, the player hand would only be detected within a small range of distance from the body. The phone also had to be held closer to the face than one would normally hold a phone. This would significantly impact the range of motion while playing. Holding the phone to detect the hand would mean holding the screen at a distance that would be harder to interact with. Conversely, holding the phone at a comfortable distance would mean stretching one's hand further than what is comfortable.

[PICTURES]

4.1.1.2 Augmented Reality

The Niantic developer portal provides a series of tutorials that can be followed in order to test various parts of the Lightship ARDK (Niantic, 2022). Tutorials were followed to create an AR session, detect planes, place objects, and track the player focus with a 3D cursor.
The Lightship cursor feature was an effective method of interaction with the environment. From these tests it was decided that an AR cursor would be a part of the card gameplay interaction model. The AR cursor would be used to select cards and an on-screen menu would be used to display the gameplay options.

4.1.1.3 Networking

The Lightship ARDK tutorials were initially built and tested in the unity editor and then build on an iPhone 12 Mini and an iPad Mini. Testing on two live devices allowed for testing of the Lightship ARDK multiplayer netcode. Lightship has two types of networking API, a low-level API for sending byte streams between all devices connected to a session, and a high-level API for sharing network enabled class instances. It was decided that the low-level API would be used to translate the stats of each card between all devices connected to a game (Niantic, 2022).

Restrictions in the Lightship ARDK networking API were overlooked during this build. This will be covered in detail in section 4.1.3.2.

4.1.2 Build 2

4.1.2.1 Class Structure

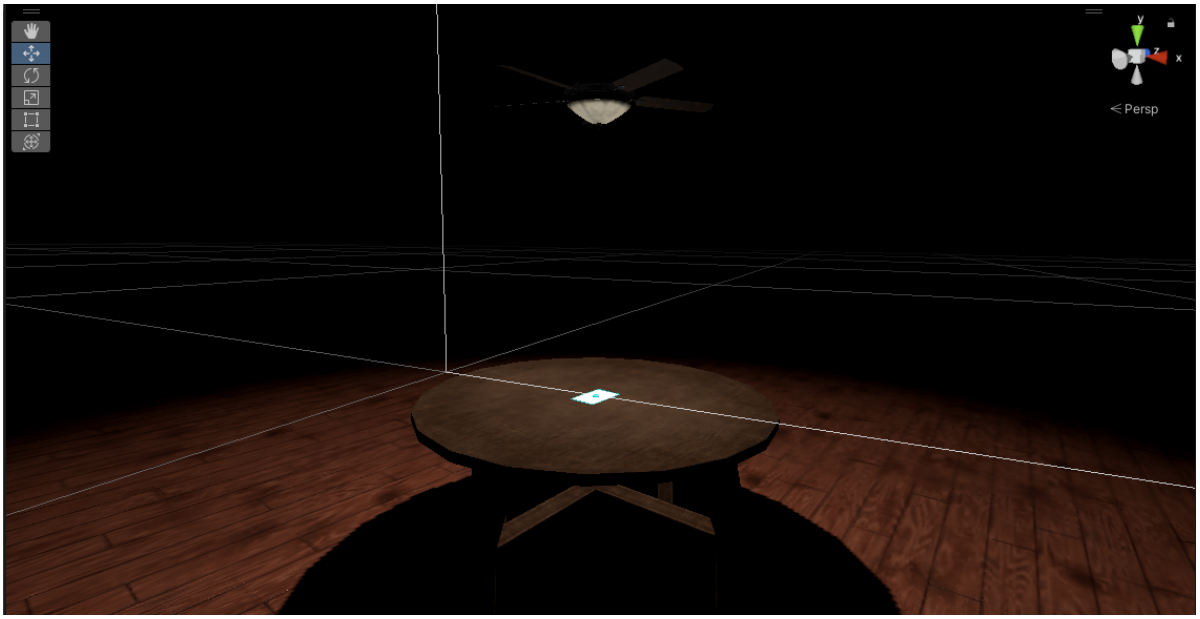
Before stepping through the how CardsAR was constructed stage by stage, the class structure will be detailed. CardsAR has a set of manager classes that are each singleton classes. Singleton classes allow for each manager to be referenced without creating multiple variables or links between managers. Other scripting is used in prefabs from third party plugins, the manager classes created for this project are as follows.

- GameManager: Controls and stores the game state.
- MenuManager: Uses the current game state information to hide and show UI elements.
- PlayerManager: Contains methods for interacting with the deck of cards from the point of view of the player. The UI buttons trigger methods inside this manager.
- MultiplayerNetworkingManager: Handles the movement of all player avatars
- EnvironmentManager: Controls how the virtual room is orientated and places in the game.
- AgoraVoiceManager: Handles the voice call.
- PubNubManager: Handles all networking data sending and receiving.
- NetworkCardManager: Stores information about all cards states. Also contains methods for modifying card data through the network:
- DeckSplitManager: Used for storing information for the pick-up multiple cards interface.

[class diagram]

4.1.2.2 Environment Setup

The virtual room created for CardsAR has a wooden table in the centre, a wooden floor, and a rotating fan light overhead. These were all taken from the Unity marketplace. The table is lit from above with singular a directional light, and outside of this spotlight is complete darkness.



When the player starts their game, they are asked to place a 3d model of a deck of cards onto a flat surface in-front of them. The placement of this card sets where the surface of the circular card table. Orientation and placement of the virtual environment was implemented without consideration for other players during build 2, the virtual room would be places in a static position relative to the deck of cards virtual marker. For a detailed description of how the environment was orientated for each player please see section 4.1.3.1.

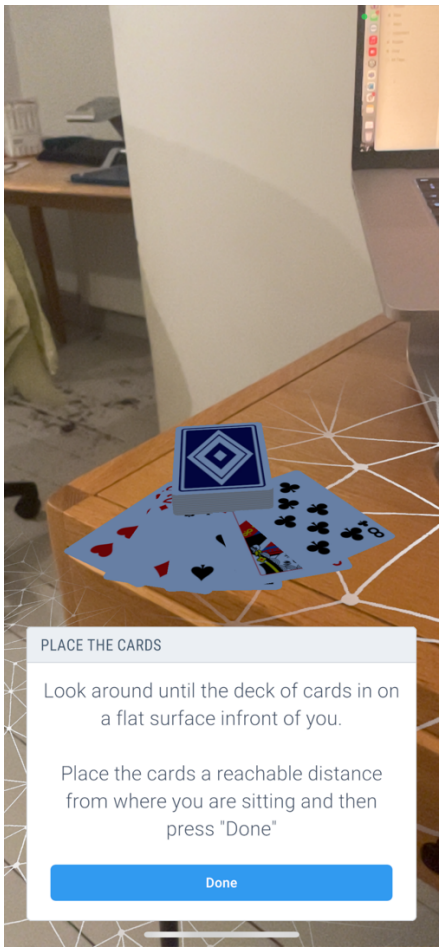


Figure d: Virtual marker placed on a table

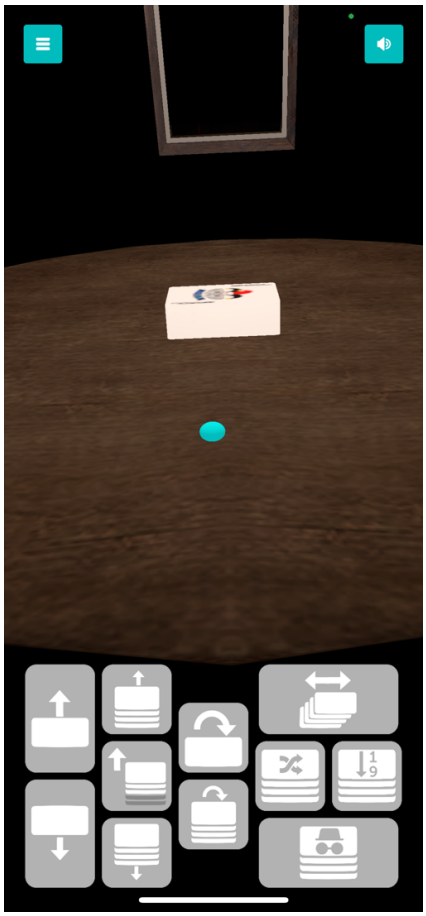


Figure e: Table orientation after virtual marker position is confirmed

4.1.2.2 Card Interaction Design

Most of the time spent on build 2 was spent on designing the card interaction model and associated UI.

4.1.2.2.1 Sticky Cards

Each card is modelled by a unity game object prefab. A prefab is a reusable game object that also allows for inheritance of properties and child game objects from a parent prefab. Each single card is its own prefab, and each of the cards inherit from a parent prefab which stores the game object data that is common to all cards.

The example below shows the prefab inheritance structure for a card that is used in the game. A card has three 3D models contained within, the playing card, a 'hidden' card which is covered in question marks and a highlight model with is rendered in one block colour. All of these are based on a playing card prefab set that was take from the unity store (Game Asset Studio, 2019). A base sticky card prefab exists for all cards used in the game and that is inherited by a prefab that exists for each individual playing card.

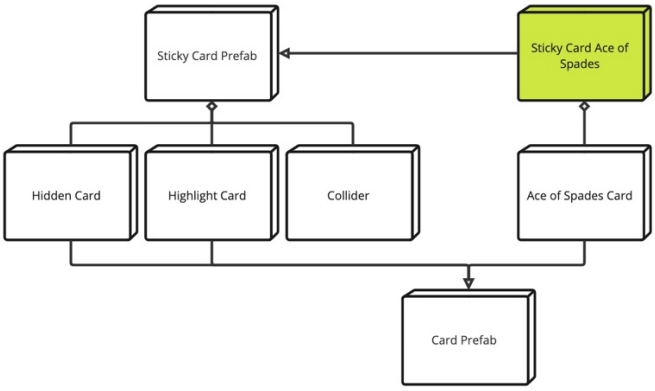


Figure f: Ace of Spades UML diagram

Attached to each card prefab is a component class called StickyCardManager. This class is the access point for all state information on a card and its relationship other cards that it is above or below in a stack. In an effort to keep the game design as simple to implement, only the cards were modelled in the game, there is no representation of deck of cards. "Sticky" is a reference to the cards ability to be linked to each other in order to form stacks of cards. When a card is sitting above another card, its position and rotation in the game world is defined by the card below it. Cards are associated together as a doubly linked list, cards can be linked to two other cards, one above and one below.

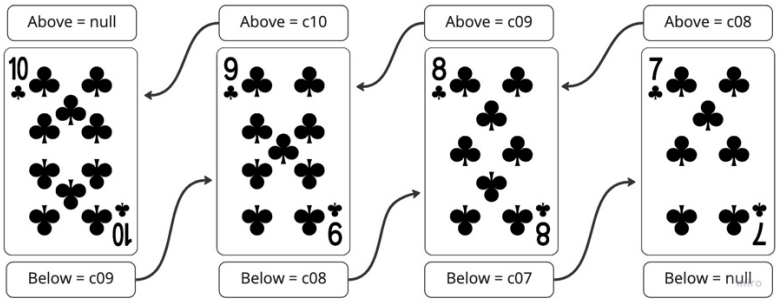


Figure g: Cards connected through as a doubly linked list

In order to cover the game states listed in section 3.3.2. cards have the following fields.

- **Position:** A 3D vector storing cards position relative to the tabletop. A zero vector is equivalent to the centre of the table.
- **Rotation:** A 3D vector s
- **Above:** A game object reference to the card above in a stack
- **Below:** A game object reference to the card below in a stack
- **FaceUp**: A Boolean value which when true the card will show the number and suit face. When false the back of the card will be shown.
- **HighlightedBy**: The unique player ID of who is looking at a card.
- **HiddenBy**: The unique player ID of who has hidden a card.

- **Spread**: This Boolean value allows for stacks of cards to be spread across the table. This allows for each card in a stack to be visible but still linked together.

As well as these state variables, the StickyCardManager has multiple methods used to retrieve information about a cards position in a stack of cards.

- **CountAbove:** Return the number of cards above the current card.
- **GetBottom:** Return the card at the bottom of the stack
- **GetTop:** Return the card at the top of the stack

Each card prefab has multiple child game objects that are changed depending on the four fields listed above. The hierarchy of a playing card game object is as follows:

- Playing Card Prefab
    - Box Collider
    - Card Group
        - Hidden Card
        - Card Highlight
        - Playing Card

The box collider is double the width and height of the playing card mesh. An enlarged collider makes it easier to select cards and helps keep cards separated on the table. For example, if a player is dropping their held card onto the table, they need to look to a table position that is outside another cards collider to drop it onto the table. This helps prevent individual cards from overlapping.

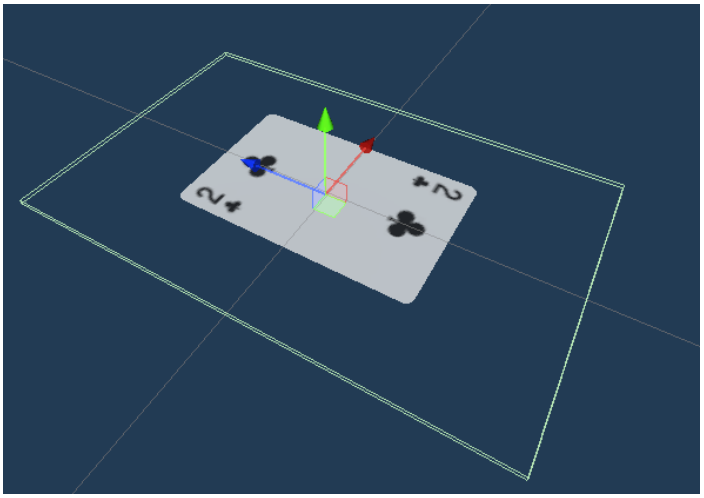
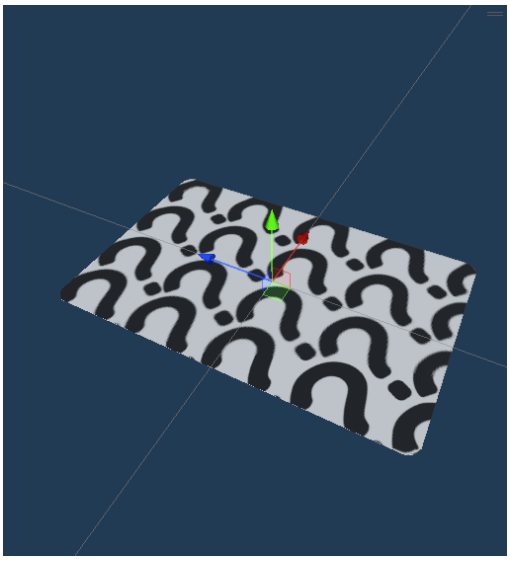
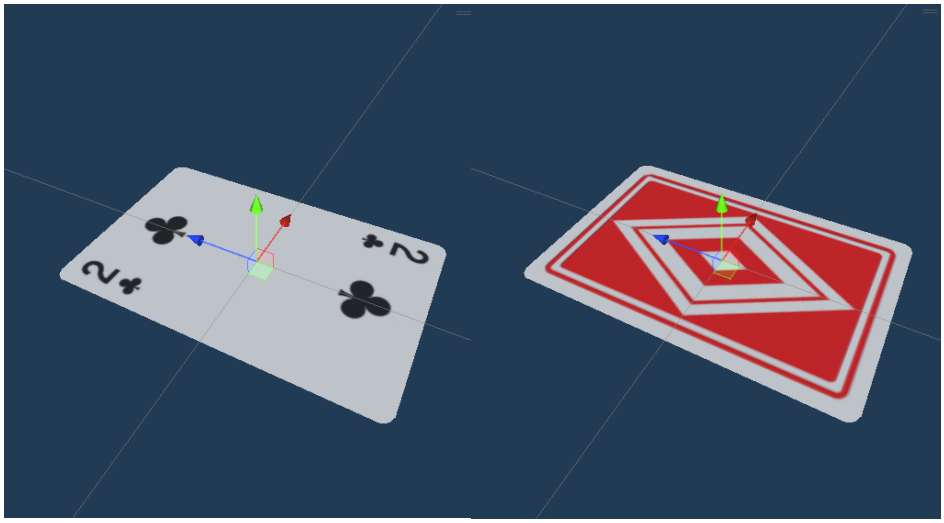


Figure h Card Collider

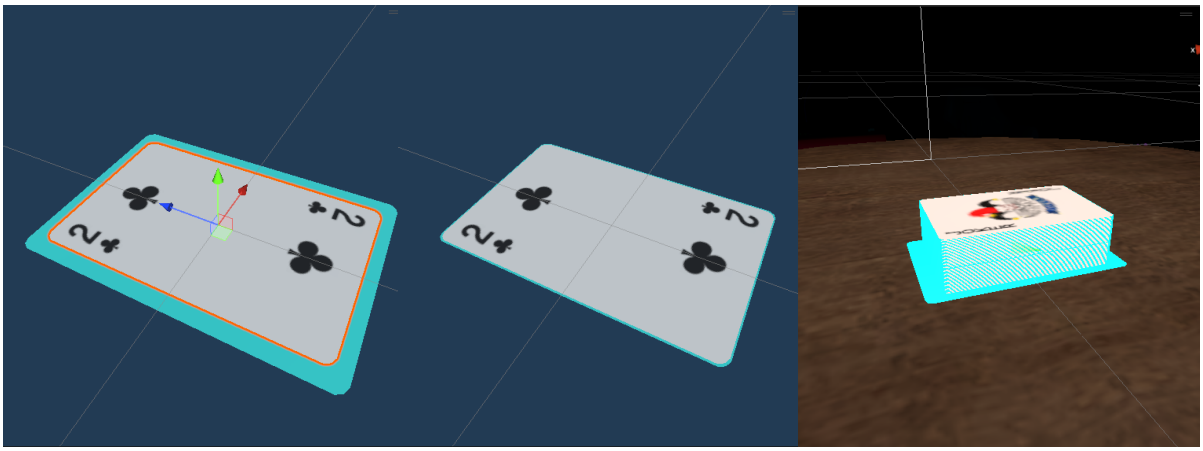
The hidden card is a card mesh with a texture of question marks.



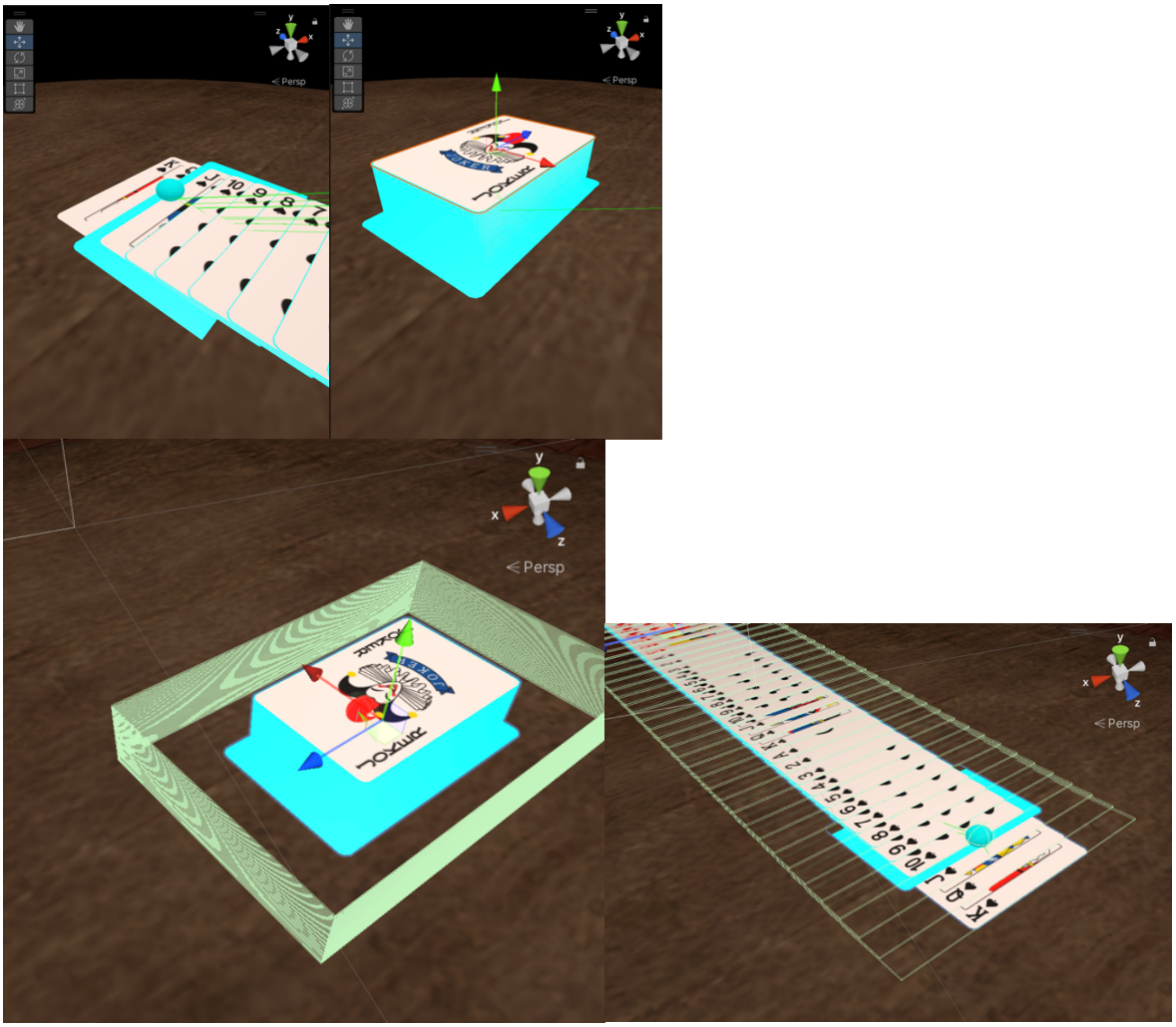
The playing card is a mesh with two textures. Whenever a card has its face up value set to false. The playing card mesh is inverted by setting the Y value of its transform to -1 instead of 1.



The card highlight is a card mesh with a solid colour texture. The highlight changed size depending on its highlight status. Initially the highlight object size is 0, but if the card prefab is highlighted, the highlight object size set to 120%. If the card is connected in a stack to another card below which is highlighted, the size is set to 102%. This smaller highlight allows for stacks of cards to be highlighted and the stack height.



The position and rotation for each card depends on its position in s stack of cards. If the card had no other card below it, the position and rotation stored in the StickyCardManager are used in the game object transform. If the there is a card below the current card the card below position and rotation is used. This would place all cards in a stack inside each other. Therefore an additional transform vector is applied to each card in the stack. If the stack is not spread the additional vector is upwards in the Y axis. If the cards are spread the extra transform is sideways perpendicular to the rotation of the base card. Spread cards are rotated so that they can naturally overlap, and their collider is ed adjusted so that no colliders overlap.



4.1.2.2.2 Network Card Data

The Sticky Card data needs to be synchronised across other players in CardsAR. How that networking layer is implemented will be handled in section 4.1.3.2 , but for now the card data is be stored in a separate class called NetworkedCardData. Each of the 52 playing cards has an associated NetworkedCardData object that is stored by the singleton manager class NetworkManager. Each instance of the StickyCardmanager class referes to the NetworkCardManager when retrieving information about card states.

4.1.2.2.3 Card Highlighting

Inside the unity scene a game object used as a cursor. Raycasts are directed out from the main camera, and if they collide with the tabletop the cursor is moved to the location of the hit.

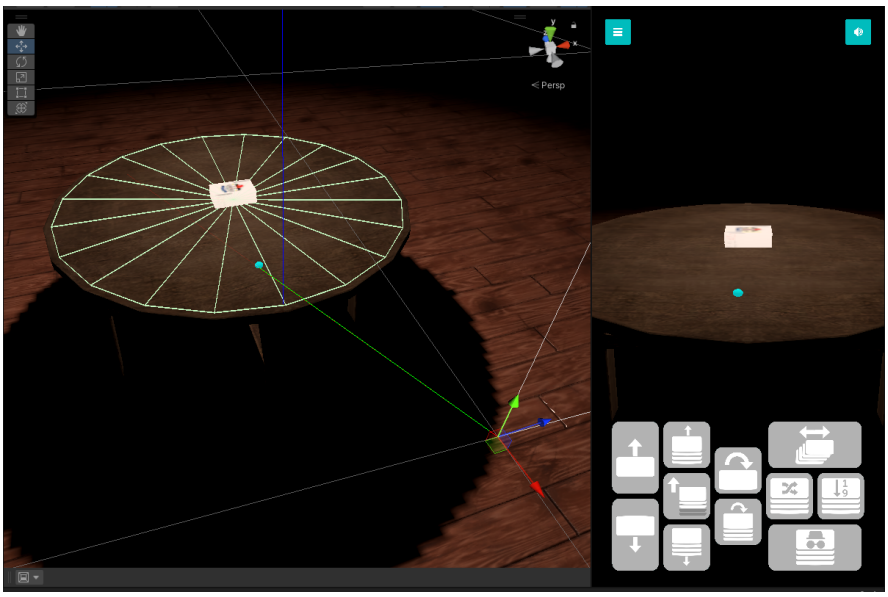


Figure i: A ray being drawn from the camera to where the player is looking on the table. Setting the cursor position.

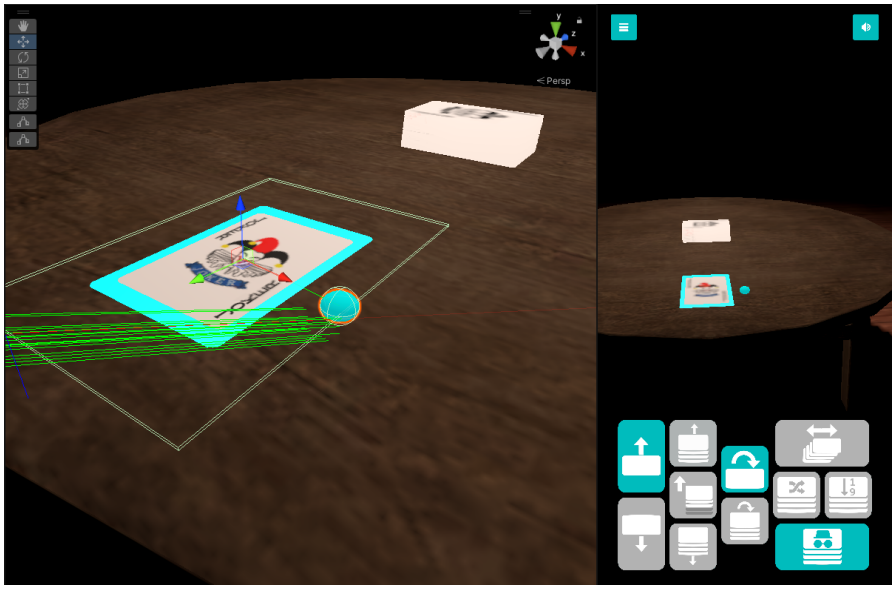


Figure j: The cursor collider crossing the card collider casung the card to be highlighted.

The cursor has sphere collider that triggers a method call whenever it enters any of the card colliders shown in section 4.1.2.2.1. This method call adds the card linked to the current collider to a list on the PlayerManager class called HighlighedCardCanditates. With every update cycle,

the PlayerManager class loops round all of the card game objects in this list and chooses the closest card to set as the player highlighted card. Cards are not added to the HighlighedCardCanditates list if they are highlighted by another player.

4.1.2.2.3 Game States

The singleton class GameManager stores the game state for CardsAR. This State impacts what parts of the UI and the sates of those individual components. This includes the introductory menu as well as the different UI states for card gameplay. The non-gameplay states are:

- **SettingDisplayName**: This is initial game state input box for entering the players name
- **ARResetInfo**: This state shows a in info pane that describes the how to best to play the game.
- **ARReseting**: During this state the player places the deck of cards virtual AR marker in on a detected AR plain.
- **ARLocked:** This game state is triggered when the player confirms the placement of the AR marker table. This state is immediately followed by the main card gameplay loop.

During the main gameplay loop the game state will be defined by how many cards the player is looking at or holding. A player can hold zero, one or a stack of cards. Simultaneously a player can be highlighting zero, one or a stack of cards that might be in a spread state. All of those conditions combined create 12 states.

- NoneHighlighted_NoneHeld
- NoneHighlighted_OneHeld
- NoneHighlighted_ManyHeld
- OneHighlighted_NoneHeld
- OneHighlighted_OneHeld
- OneHighlighted_ManyHeld
- ManyHighlighted_NoneHeld
- ManyHighlighted_OneHeld
- ManyHighlighted_ManyHeld
- SpreadHighlighted_NoneHeld
- SpreadHighlighted_OneHeld
- SpreadHighlighted_ManyHeld

[State diagram]

4.1.2.2.4 Card UI

The UI buttons for CardsAR have are designed from the users point of view. The actions a player could want were identified as the following.

- Pick up a card
- Put down a card
- Pick up a stack
- Drop a stack
- Pick up several cards.

- Turn a card face up
- Turn a card face down
- Turn a stack of cards face up
- Turn a stack of cards face down
- Spread a stack of cards
- Stack a spread of cards
- Sort a stack
- Shuffle a stack
- Set a visible card or stack to hidden
- Set a hidden card or stack to visible
- Set a visible stack to hidden
- Set a hidden stack to visible

Several options were considered by eventually removed as they were not necessary for most games, and their functionally could be achieved from multiple actions with other buttons:

- Pick a card from a random position in a stack
- Insert a card to the middle of a stack
- Pick up a card from the bottom of a stack
- Drop a card to the bottom of a stack

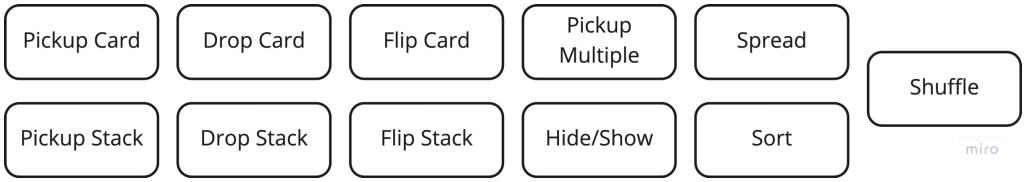
During early development for the UI, consideration was given to consolidating some UI buttons to save space. The face up/down, hide/show and spread/stack functions only need to be one button each and can function as toggle. The hidden, face up or spread state of a card should be clear to see from the players view, so the player should understand what happens when they press the button.

Functionality for face up/down on individual cards is not kept separate to the functionality for face/up on a stack. This is because players may want to turn over only the top card on a stack.

The pickup and drop card and stack functions cannot be designed as a toggle as it should be possible for a user to pick up more cards while they are holding one. This allows for quicker pickups of small number of cards. Similarly, a player should be able to hold a stack of cards and drop individual cards from the bottom. This does not allow for player to move a deck around the table dropping cards to one player at a time.

Sort and shuffle, even though they have opposite functions, cannot be grouped into one button as a player cannot tell from looking at a card stack if it sorted or shuffled.

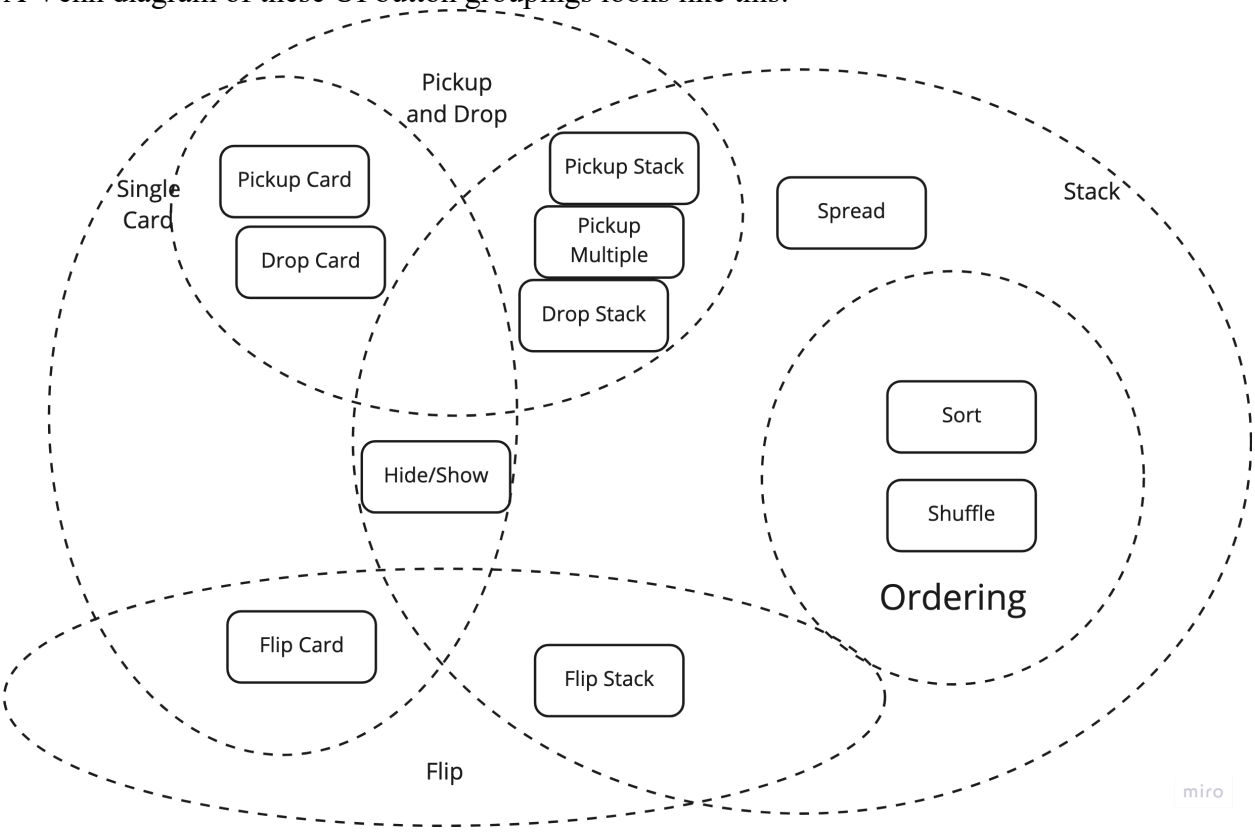
This reduces the list of user actions to 11 buttons.



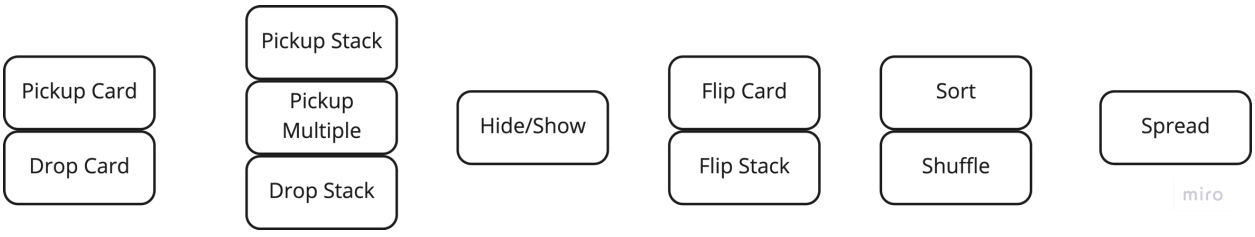
These buttons can be grouped together in ways that help with organisation of the UI. These groupings are:

- Pick up and drop functions
- Single card functions
- Stack functions
- Flip functions
- Ordering functions

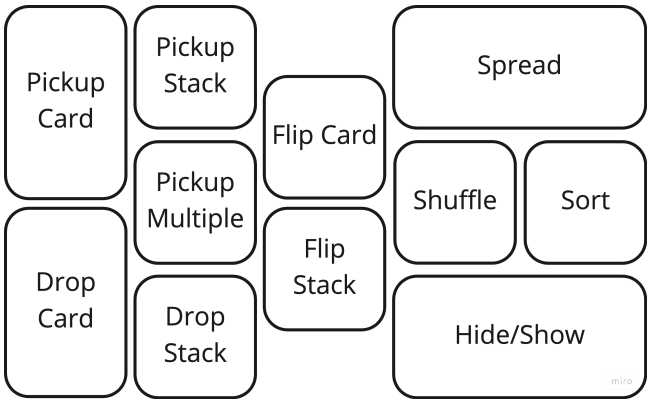
A Venn diagram of these UI button groupings looks like this:



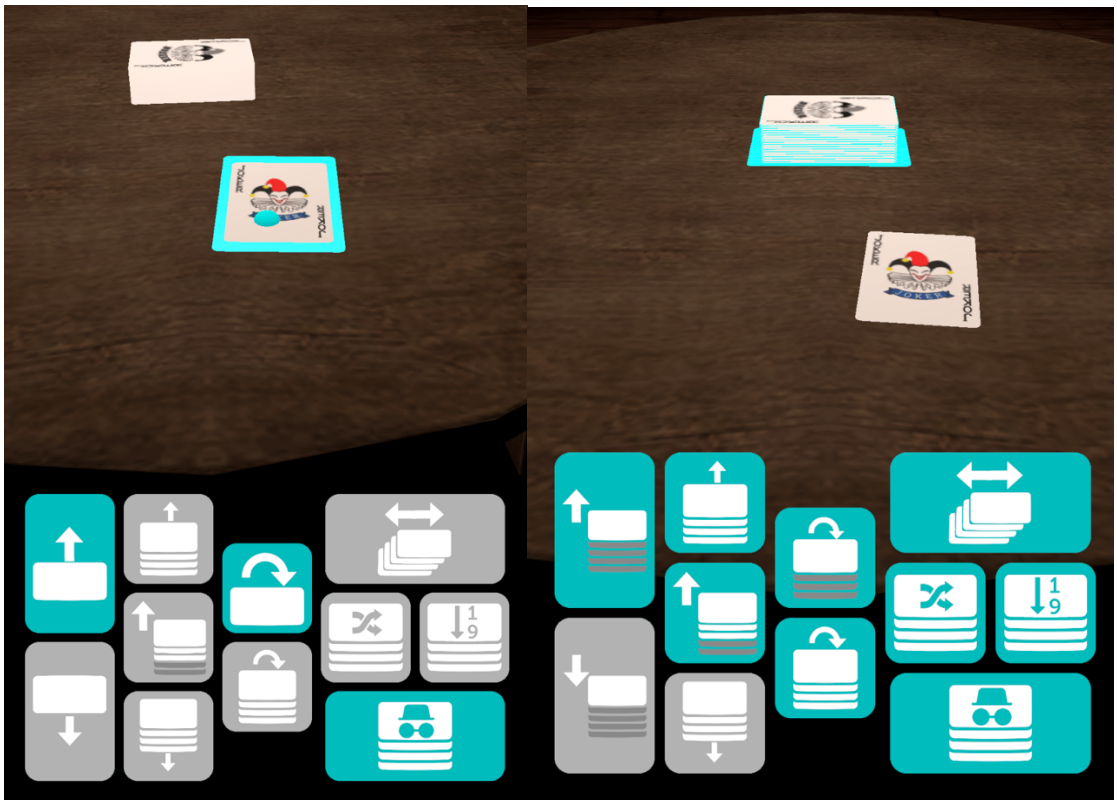
The grouping of buttons from within the digram can be maintained the UI layout in a layout similar to the following:



CardsAR is designed to be played on a smartphone held in portrait orientation. The UI sits to the bottom of the screen in a rectangular area that will scale to the resolution of the device. The final arrangement of buttons maintains most of the grouping identified above.



The arrangement is asymmetrical from left to right which was used to help with the separation of groups. Buttons that were expected to be used more were also enlarged. The final implementation of the UI used custom made icons that on some of the buttons change according to the game state.



The screenshot on the left has the player looking at a single card so the pickup card button has an icon of a single card. On the right hand side, the player is looking at a stack of cards so the pickup card icon shows the top card from a stack being selected.

Depending on the current game state the UI buttons are enabled and disabled. The following diagram shows the enabled state of each button for each game state.

Game State	Card UI Buttons										
	Pickup Card	Drop Card	Pickup Stack	Pickup Many	Drop Stack	Flip Card	Flip Deck	Spread	Shuffle	Sort	Hide
NoneHighlighted_NoneHeld											
NoneHighlighted_OneHeld		✓				✓					✓
NoneHighlighted_ManyHeld		✓			✓	✓	✓		✓	✓	✓
OneHighlighted_NoneHeld	✓					✓					✓
OneHighlighted_OneHeld	✓	✓				✓					✓
OneHighlighted_ManyHeld	✓	✓			✓	✓	✓		✓	✓	✓
ManyHighlighted_NoneHeld	✓		✓	✓		✓	✓	✓	✓	✓	✓
ManyHighlighted_OneHeld	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓
ManyHighlighted_ManyHeld	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SpreadHighlighted_NoneHeld	✓		✓	✓		✓	✓	✓	✓	✓	✓
SpreadHighlighted_OneHeld	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓
SpreadHighlighted_ManyHeld	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

4.1.2.2.5 Card Functionality

Each button when tapped, triggers a function in the PlayerManager named after the calling button name. These methods then trigger appropriate methods in the NetwordCardManager class, based on what cards are hidden or highlighted. The NetworkCardManager handles the re-arrangements of card data. These methods are:

- FlipWholeDeck(string cardID, bool pickup=false)
  - Toggle the faceup value of a deck of cards from the bottom cardID. Also reverse the order of the cards. Optionally pick up the deck immediately afterwards.
- FlipDeckOneByOne(string cardID)
  - Toggle the faceup value of a deck of cards from the bottom card's cardID. Also reverse the order of the cards. Optionally pick up the deck immediately afterwards.
- UpdateHeldPosition()
  - Use the current cursor position of update the position of any currently held cards.
- SetFaceUp(string cardID, bool value)
  - Set one card's faceUp value.
- SetSpread(string cardID, bool value)
  - Set one card's faceUp value.
- ShuffleDeck(string cardID, bool pickup=false)
  - Randomly order a deck of cards from the bottom card's cardID.
- MagnetSingle(string heldCardID, string highlightedCardID)
  - Pick up the highlighted card and add the previously held card to the top of the newly held card.
- SinglePickUp(string cardID)

    - Pick up one card.
- MagnetDeck(string heldCardID, string bottomCardID, string topCardID)
    - Pick up the deck from the bottom card and add the previously held card to the top of the top card.
- DeckPickup(string cardID)
    - Pick up a card deck.
- DropOneCardOnTopOf(string thisCardID, string thatCardID)
    - Put this card on top of that card.
- DropOneCardOntoTable(string cardID)
    - Drop the card onto the current location of the cursor.
- DropDeckOnTopOf(string deckBottomID, string deckTopID, string highlightedID)
    - Drop the deck bottom card on top of the highlighted card.
- DropDeckOntoTable(string cardID)
    - Drop the current deck onto the current location of the cursor.
- SetHideDeckTo(string cardID, string value)
    - Set a deck's HiddenBy value from the bottom card of a deck.
- SetHideCardTo(string cardID, string value)
    - Set a card's HiddenBy value.

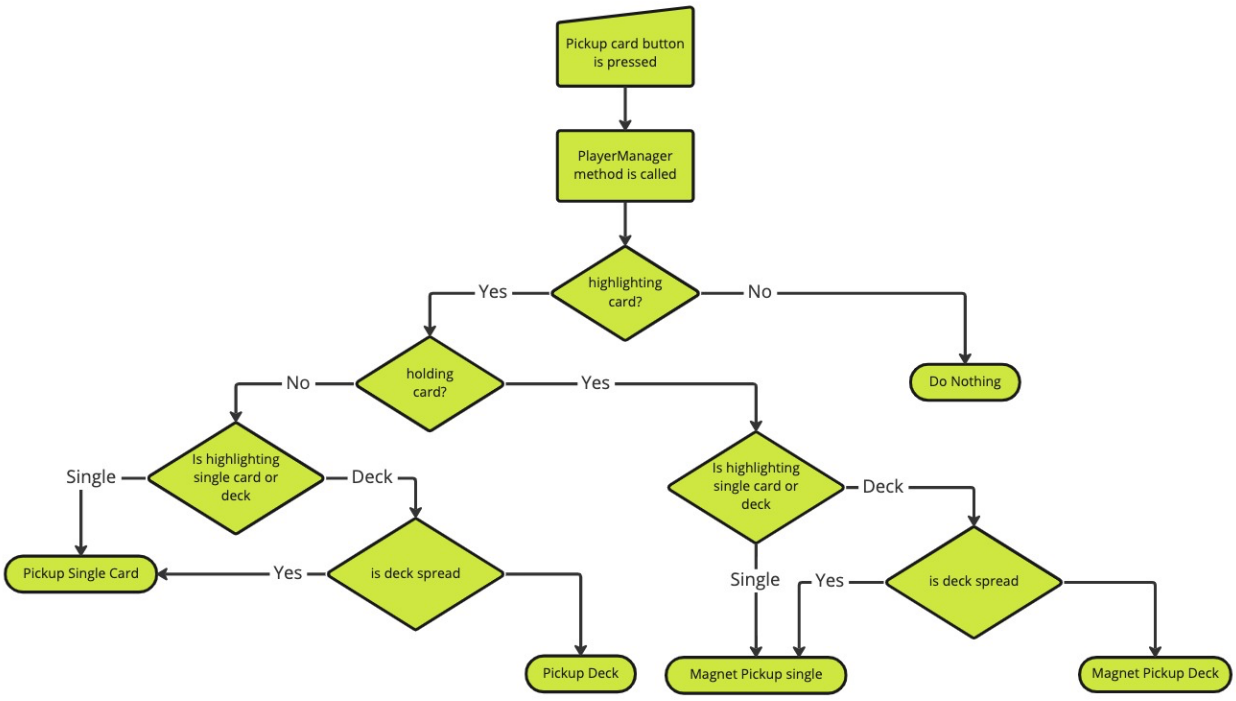


Figure k: Example flowchart showing the decision tree for selecting the correct NetworkCardManager method

4.1.3 Build 3

4.1.3.1 Avatar Design

The integrated video calling for CardsAR was designed to be rendered into the 3D virtual room. The orientation and position of each players device would be tracked and used to move around an avatar representing the player. A picture frame model was taken from the Unity marketplace [CITATION] and used as the basis for a player avatar. The player's name is entered at the start of the game and displayed at the top of the avatar during gameplay.

Player orientation around the table is important for gameplay purposes. The class EnvironmentManager handles the placement of the current players table, as well as placing each players avatar around the game table. The aim is to have each player equally place around the table, and to have their relative positions to the table and to each other consistent across all players.

For each player in the game, a deal position is created. A deal position is a 3d coordinate point on the table that is equally spaced from the other deal positions for other players. When arranging the table positions consistency between game sessions in achieved by using each players ID in sorted list.

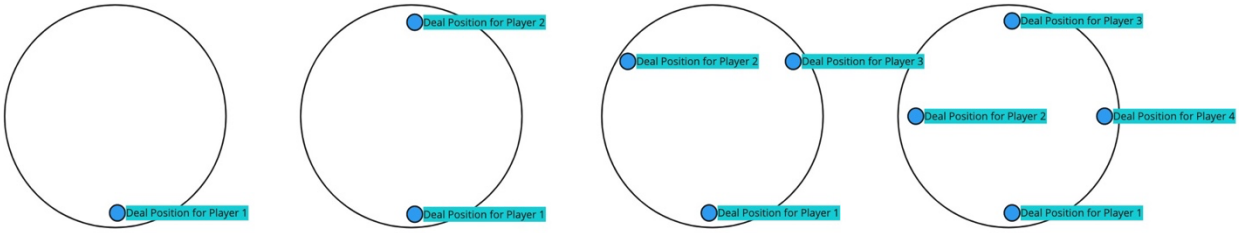


Figure l: Deal postions as seen from above for 1-4 players

At the beginning of the game the player confirms the AR position of the table. They do this by placing a 3D stack of cards on a surface in their real-world environment. In the EnvironmentManager class this is referred to as the TablePlacer as it used to set the position of the virtual table. When they press the confirm button, the current TablePlacer positon is saved to

a variable called LockedTablePlacerPos, along with variables for the camera's locked postion rotation and forward vector. These values are then used to orientate the virtual world and lock the rotation of the table according relative to the current players deal position.

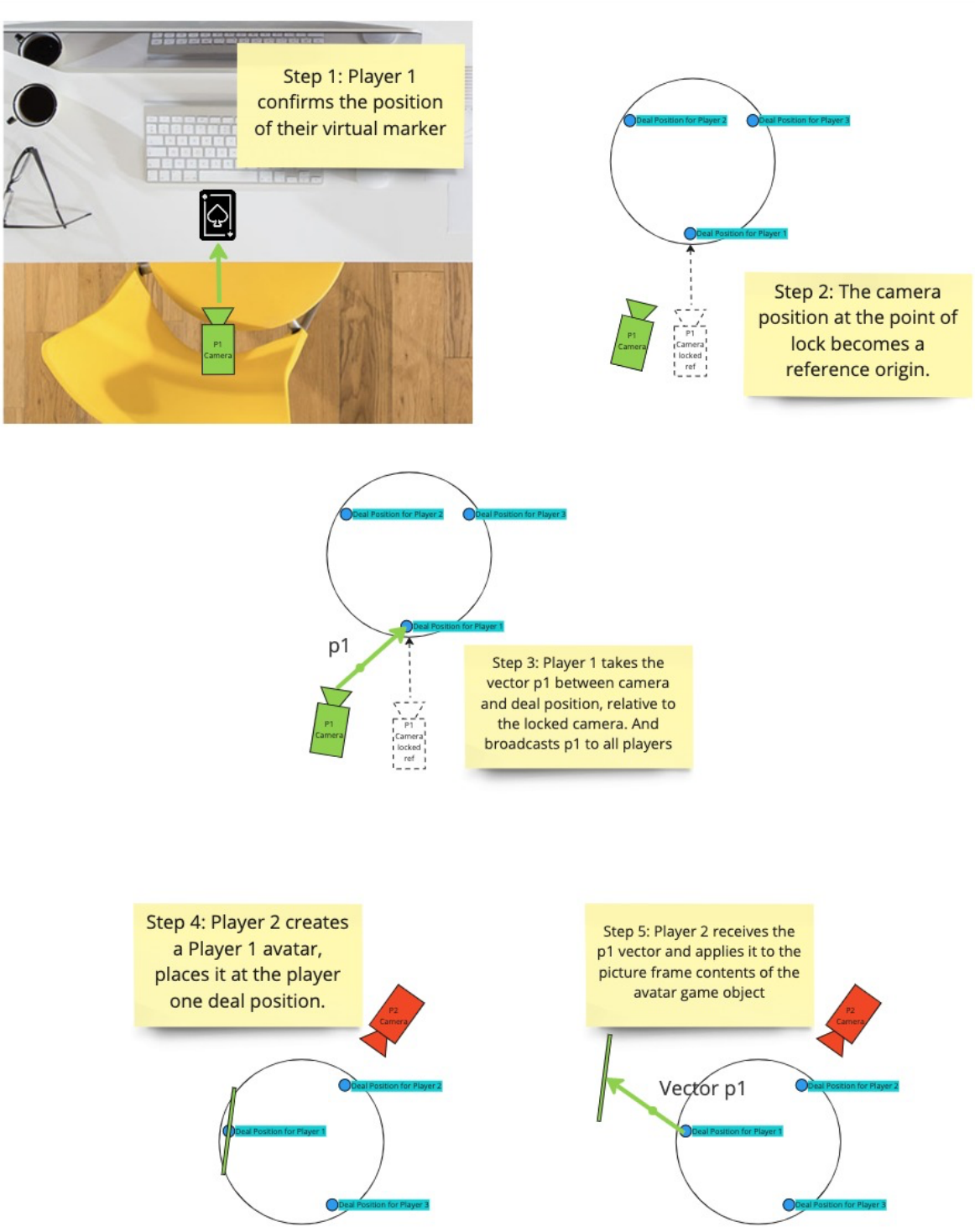


Figure m: How the player movements are tracked, sent to other players and added to avatar game objects

4.1.3.2 Netcode Redesign

Two types of data need to be synchronised across all players, the card data and the position and orientation of player avatars. Originally the plan was to take advantage of Lightship ARDK multiplayer session management and the low-level networking API to send byte streams of data between devices. This was tested in build one, but a critical component of the Niantic multiplayer model was overlooked.

Lightship ARDK networking is contingent on players sharing the same physical environment. The host player in a networking session automatically scans for visually distinct items in the environment to create virtual markers from. These markers are then searched for by other players in the session and only after the same virtual markers are found and synchronised, does the networking session become useable. This was overlooked during build one as the environment syncing is automatic process with the Lightship ARDK prefabs. Testing was also done on two devices in the same physical space, so the local environment was quickly synchronised without issue. During build three testing was continued on the same devices, but the tests were done in by two people in separate rooms, which immediately highlighted the limitations of Lightship ARDK. A support ticket was raised with Niantic who confirmed that shared AR experiences are required to take place in the same physical space (See appendices B.1: Niantic).

Losing Lightship ARDK for multiplayer as a significant issue for the development of CardAR. Lightship handled the session management, data hosting, and networking protocol. There are multiple options available for Unity. Multiple options were investigated including in Unity package: “Netcode for Game Objects” as well as the third party package “Mirror” (Unity, 2020).

Time was also a factor in decision making for a new netcode solution. CardsAR had thus far been designed with the expectation that a low-level API that allowed for the passing of byte streams between players was going to be used. If more time allowed using a solution that required redesigning the game objects to be network enabled would have been the best choice. Unfortunately, as it was more important to find a quick solution, a different service was used.

PubNub is a realtime communications platform that provides a messaging API that can be used in applications such as chat rooms or internet of things devices. Using a chatroom style API was the best option for the netcode redesign of CardsAR as it would allow easy broadcasting of interaction updates from each player.

All the other networking solutions required a separate service for session and data hosting. PubNub provided free hosting and messaging for two months. The initial free trial period was for one month, but the support team at PubNub extended the free trail for an extra month after it was appealed (see appendices B.3: PubNub).

The broad networking model follows these steps:

1. A Player proposes a change to the shared game data.
2. That change is broadcast to all players.
3. All players receive that change request.
4. The game data is changed.

This networking model was the most straightforward model to implement in the time allocated. But it has several vulnerabilities that had to be worked around when coding the solution.

Card data can fall out of sync if two players attempt to change the same data at the same time. A workaround was achieved by only letting one player highlight a card at any one time. The card highlighting is also updated regularly so that two players cannot hold the same card at the same time. If two players are holding a card it will only persist until the next update cycle.

Card data is also only changed through the PubNub data manager. This helps all card data stay synchronised across all players. Unfortunately, this also adds input lag in-between a player pressing a button and seeing the resulting change.

The broadcast model also makes persistent data difficult to share. Some data, such as the position of player avatars need to be constantly updated to all players. Broadcasts a constant stream of this data in an inefficient use of bandwidth. A player does not need to process each avatar movement as it is sent, only the recent avatar position is important. This was worked around be limiting the number of avatar updates to one every 0.5 seconds. This led to stutter-y player movement, but it avoided overloading the PubNub channel with messages.

All communication is handled through the PubNubManager class. This class contains hardcoded authentication and session information so that it can connect to an PubNub channel that has been created for the CardsAR project. After successfully connecting to the Pub Nub service a callback is created for responding to any messages added to the channel.

Messages sent into the PubNub channel are dictionaries with string keys and string values. Each message has a type key which is one of seven different types of message that can be sent.

- PLAYER_HAS_CHANGED_POSITION
    - These messages are for avatar movement changes, they contain the player's name, ID, avatar position and rotation.
- PLAYER_HAS_CHANGED_DECK_DATA
    - These messages contain a list of card data strings which as detailed in the next section.
- PLAYER_HIGHLIGHT_START
    - Contains a card ID and player id to set the highlighedBy variable of a card.
- PLAYER_HIGHLIGHT_STOP
    - Contains the card ID to clear the highlighedBy variable of a card.
- PLAYER_HELD_START
    - Contains a card ID and player id to set the heldBy variable of a card.
- PLAYER_HELD_STOP
    - Contains the card ID to clear the heldBy variable of a card.
- PLAYER_REQUESTING_HOST_UPDATE
    - This is an empty request but when the "host" receives this message the host broadcasts the state of all cards.

The player position change, and highlight changes, and changes to the position of a held card are broadcast every 0.5 seconds. Without this limit the PubNub channel would be overwhelmed by

messages. This is controlled a C# invoke call which is set up in the start call of the PubNub manager.

All other PubNub messages are triggered by the NetworkCardData manager that is covered in section 4.1.2.2.2. When buttons are pressed the PlayerManager calls the NetworkCardData manager who translates the cards modifications into a list of NetworkCard change strings.
A change string is serialised form of the card data. A player generates change strings and sends them though the PubNub channel. Every player then reads those change stings and applies the changes in data to their card data. A change sting has the following format:

```
"{faceUp}/{spread}/{position.x}/{position.y}/{position.z}/{rotation.x}/{rotation.
y}/{rotation.z}/{cardBelowID}/{cardAboveID}/{hiddenBy}"
```

For example a card change string may look like this:

```
"true/false/1.0/0.0/3.0/0.0/135.0/0.0/a2/a4/00001"
```

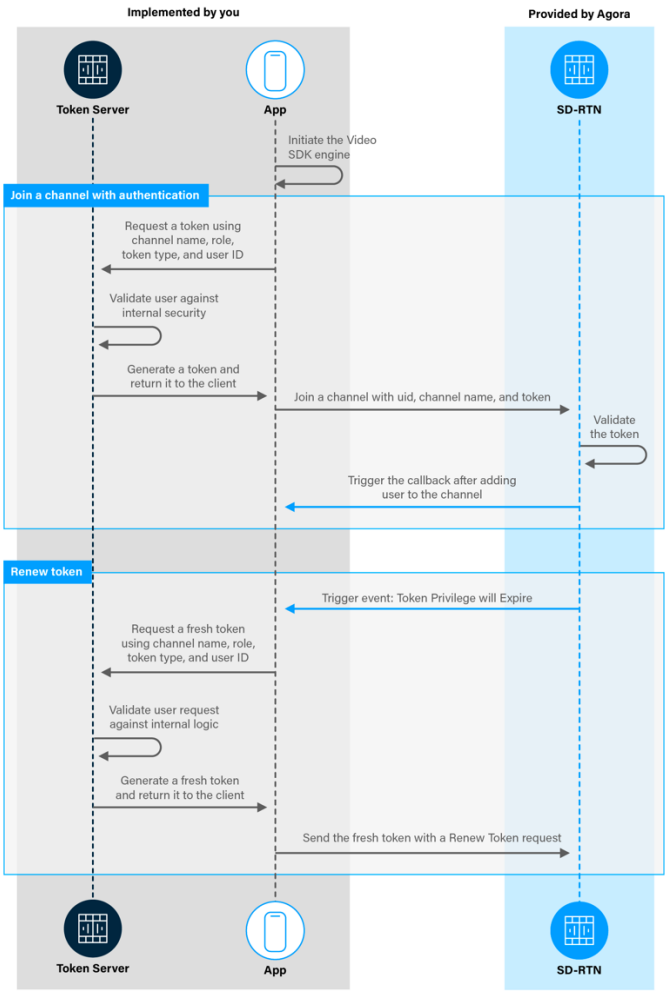
From The UI to the card data changing, the following diagram shows how data is passed between the manager classes.

[diagram of network data]

4.1.3.3 Video Call and Audio Call

In addition to the difficulties encountered with multiplayer, video calling proved to be too difficult to integrate. The Agora video SDK was tested in build one was tested in isolation, but it wasn't until build three that it was discovered that video calling disabled the world facing camera, which disabled AR. A support ticket was raised with the Agora software support team who confirm that even though apple have supported using the video feeds from both front and rear facing cameras, that Agora SDK had not been updated to support that feature (see appendices B.2: Agora). It was suggested by the support staff that it could be possible through writing custom wrappers to the iOS camera libraries. There was not enough time left to develop this feature, so the Agora voice SDK was used instead.

The authentication and call channel information it's hard coded into CardsAR. Once the player confirms the position of the table, they are automatically entered into a call. The authentication and channel information are hardcoded into CardsAR. To join a call an authentication server has been deployed using Railway, this authentication server is used to generate tokens for use in the Agora API. The token server deployment and API integration into Unity was achieved through the guides provided by Agora (Agora.io, 2022). The following diagram from the Agora website shows how authentication with the audio call works.



A small mute button was added to the right-hand side of the interface to allow players to mute the audio call during the game.

4.2 Finished Game

4.2.1 In Game Screenshots

4.2.2 Video

4.3 Testing

4.3.1 Testing during development

Every new feature was play tested during development. Initially features were tested within the unity interface using the Lightship virtual studio. This feature allowed for AR features to be tested without building to the test devices.

When testing for multiplayer, multiple dummy players were spoofed and their movements were tracked to movement of player one.

Multiplayer netcode stability was tested with three players, the Unity game window, an iPhone Mini, and an iPad window. As this was the internal testing limits, external testing was limited to three players.

Informal testing of the interface was also done with the authors partner. Who helped highlight the redundancy of buttons to pick up and drop cards from the middle and bottom of a deck of cards.

4.3.2 Known Issues

Not all bugs were solved before formal testing had begun. The following is a list of bugs that were present during testing, but were small enough to continue with.

- Sometimes a card would become attached to the cursor location. To fix around this another card would ned to be picked up.
- Inside the last current build of the game the Ace of clubs game object would float higher off the surface of the table. Therefore at the begging of testing it was removed from the deck by the host.
- The cursor is intended to be locked in table surface but moving the cursor to the edge of the screen occasionally causes the cursor to float upwards off the table. If a card is dropped while the cursor is in the air, the card will not be able to be picked up again.
- Card positions are not being synced for players who have newly joined the session.
- If a fourth person joins the game, their avatar is displayed as upside down. This only happens with groups of 4. Testing was limited to groups of 3 so this bug did not cause significant problems.
- Sometimes two cards when placed in a spread, would not be correctly rotated and they would clip through each other. This could be fixed by toggling the spread button on the cards.

4.3.3 Formal Testing

The formal testing took place remotely. CardsAR was shared with volunteer testers using Apples Test Flight application. A build was uploaded using Xcode and distributed to all testers.

All volunteers were sent a link to the website zcal.com to arrange a preferable date and time. After they had submitted their preferred times they were all grouped into testing sessions of 1-2 volunteers. The author was present in all testing sessions so the groups ranged from 2-3 players.

Each testing session followed the same structure, all players would join the game at the agreed upon time, the mechanics of CardsAR were explained, and then the group would attempt to play Shithead. Shithead is a game where players attempt to get rid of all the cards in their hand (Pagat, 2021). Each player has three stacks of two cards in front of them. The top card is face up and the card below is face down. They also have five cards in their "hand". This game was chosen as it would force players to use pick up and arrange cards multiple stacks of cards as well has move cards to and from a hidden spread of cards

The game was player for around 30-40 minuets. After the game had concluded the players were asked to fill in an anonymous form for feedback. The form was created using Google forms, and it was configured to not store the players email. It was designed to gather information about the following:

- What was the players familiarity with online games.
- How did the COVID 19 lockdowns impact their social gaming habits.
- How successful was the AR experience?
- How successful was the playing cards interface?
- How much did the players enjoy the experience?

The full questionnaire and responses from all 16 volunteer testers is included in the appendices.

Chapter 5 – Discussion

Over the course of two weeks 9 testing sessions were held with 16 participants. In each session the author hosted and was able to observer each how each player interacted with CardsAR.

5.1 Author's Observations

All testers were unfamiliar CardsAR before playing. Most were unaware that it was an augmented reality game until they opened the application. When players first noticed that they were in a virtual world with each other the most common reaction was amusement. The avatar movement appeared to feature that users enjoyed the most.

Most players appeared to struggle to keep their cards organised in front of them. The play surface appeared to be too small, and it was common for half of the playtime to be spent arranging the three sets of cards in front of themselves.

The Most players made the same common mistakes when playing the game:

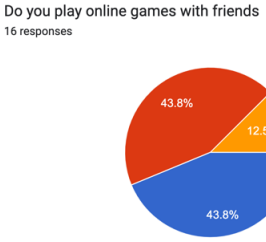
- When dropping cards onto the table players would accidentally drop them on top of other cards accidentally.
- When holding a deck of cards, players would intend to drop the whole deck, instead they would drop one card from the bottom of the deck.
- Players would forget to either hide or reveal their own cards.
- Players would "lose cards" by forgetting how many cards were under each deck.

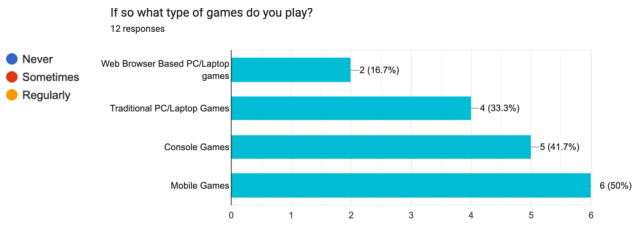
As well as these mistakes made during gameplay, the cursor system made it was impossible to look at a card without highlighting it. This meant that very often players interrupted each other's actions by highlighted cards accidentally.

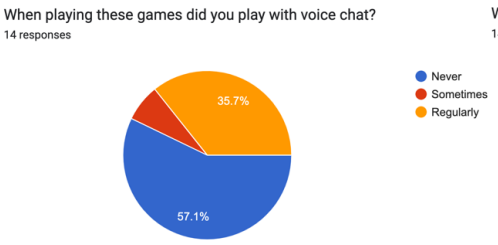
In three gameplay sessions the audio call quality dropped significantly. This meant that for a few seconds at a time, players would not be able to hear each other. As voice was the only form of communication between players, gameplay would come to a halt.

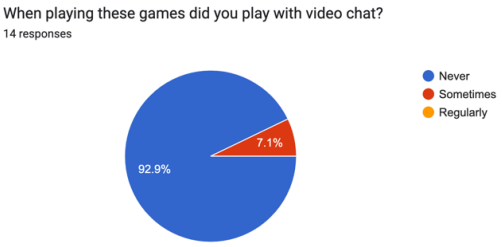
5.2 Play Tester Feedback

5.2.1 Tester Survey

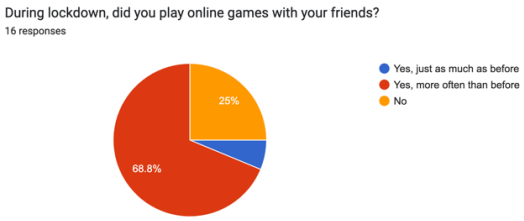


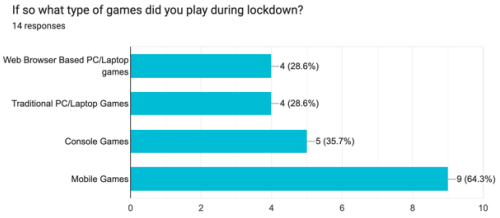


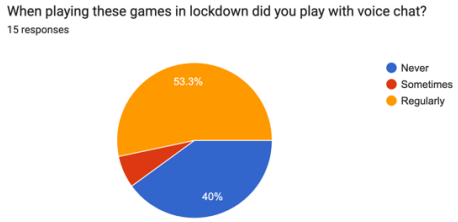


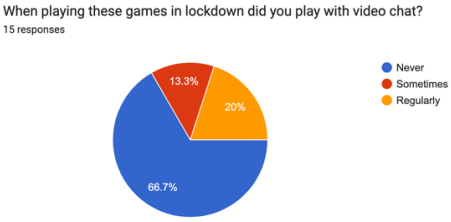


The first section of the feedback form was designed to identify the broad preferences in social gaming for testing group. From these results it can be seen that social gaming with voice chat popular with a minority of players but social gaming with video almost never used.



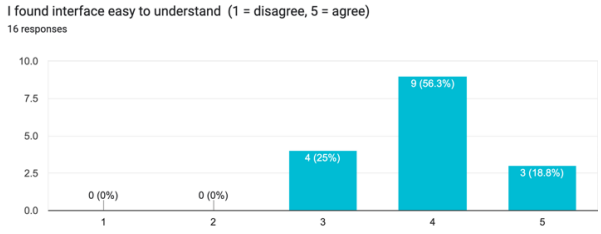


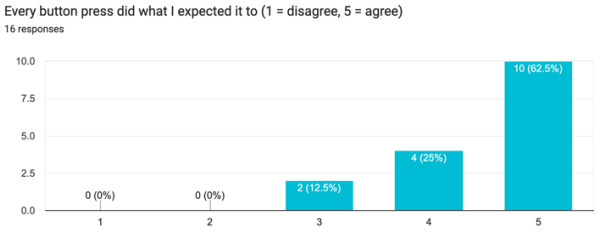


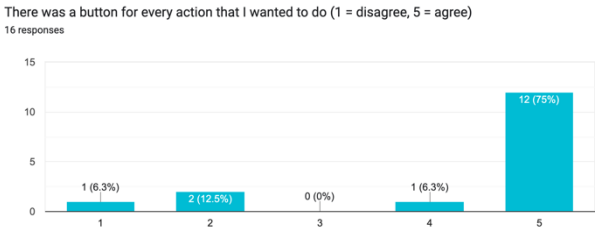


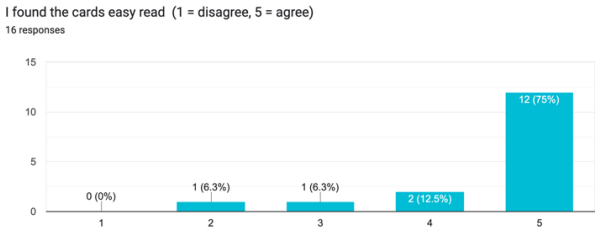
During lockdown social gaming increased with 75% the testing cohort. Smartphone based gaming was the most popular before and after lockdown, but during lockdown it was significantly more popular that other gaming mediums. Lockdown also increased the amount of video and voice chats. Noticeably, 33% of the group use video calling to varying degrees of frequency which is a 4x increase on the video calling activity outside of lockdown.

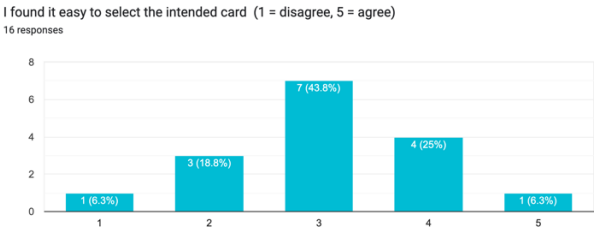
5.2.2 Gameplay Feedback

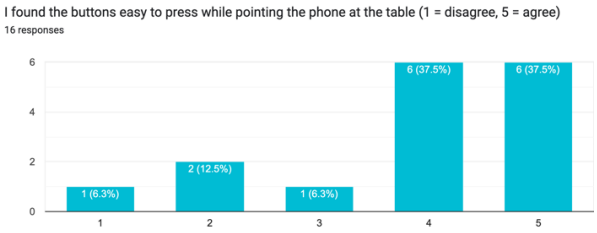


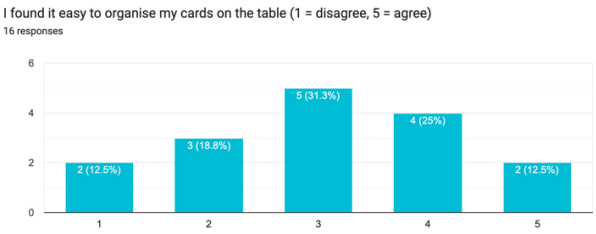












The next section of the feedback form was intended to gauge how easy to use the testers found the card gameplay model and UI.

Feedback showed that players understood the interface once it had been explained to them and they felt like the interface was sufficient for what was being asked of them. Many users commented on a small learning curve after starting the game for the first time and that they understood the interface once it had been explained to them. Some users also commented on the input lag negatively impacting their experience. Both the input lag and onboarding would hopefully be more refined in a commercially released version of CardsAR, as a proof of concept these are acceptable compromises.

"The interface was easy to understand with an explanation to say what every button was. It might have been a bit overwhelming to look at if I was just on my own."

"Can infer from most of the pictures what they do, most needed an explanation however."

"The button presses would result in an accurate action, such as flipping cards, individually picking some out, spreading cards etc. The only issue was that the lag made the button presses a bit slower."

Some players commented that the sorting features of the game were insufficient. The current sort orders by suit and then by card number, which wasn't helpful when attempting to select cards of the same number but different suits.

"The only action that was missing was sorting in order of value of card, the sort button sorted from A-K in suites, so if I had 2 aces for example they could be separated by a number of cards which isn't that helpful when playing a game based on pairs/three of a kind etc"

Questions 4 and 6 above highlight that arranging cards on the table was the most significant issue that players had. Selecting the appropriate card, dropping it in the correct place and hiding cards were all features that full succeed with testers.

"The hardest bit perhaps was putting my cards in the right place on the table (lining them up straight etc)."

"I felt my actions seemed clumsy perhaps having more room to place the cards would help"

"I found it more difficult sometimes to pick the correct card (when trying to pick up one, or various, or hide the cards for example). Sometimes it took a few tries. It could also be trickier to put the cards down on the deck in the right place"

This feeling of "clumsiness" is the result of a combination of factors. When dropping cards, it is not clear where they would be placed. It was left to the user to assume that if another card was highlighted, a dropped card would appear on that card. A reasonable fix to this could be a rectangular silhouette on the table that showed when a card would be placed.

During informal testing it was decided that the table was too large, and the cards were too small to be useable. It appears that in attempting to correct this issue, the final table space was too small. This motivated to players to arrange their cards more often, which highlighted the shortcomings of card placement.

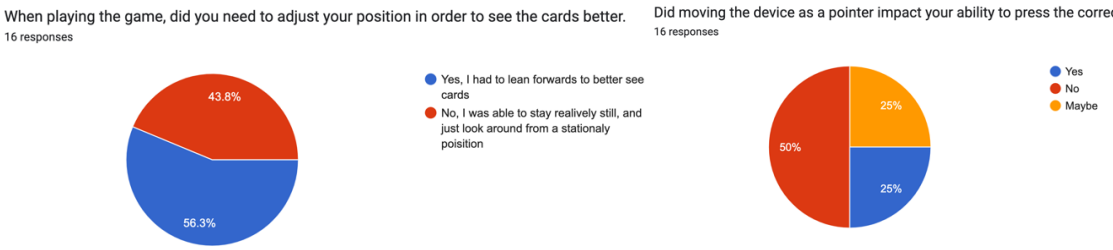
Even though cards were scaled up players still had difficulty quickly pointing and selecting cards. This could be a as result of the hastily implemented netcode, but it is also possible that cards should have been larger. If cards were larger it could have meant less space for playing, which points to a potential conflict between gameplay area and card size.

The hiding functionality did not operate as players often expected it to. Players would remove a card from a hidden deck and expect it to become visible to all players. Conversely if a player dropped a visible card into a hidden deck it would not become hidden as they expected.

"Hiding cards had a similar issue in that putting a card into my hidden deck wouldn't automatically hide it."

CardsAR was inspired by the PC and Mac game Table Top Simulator. That games approach to hidden cards is to place the cards at the edge of the table in a zone dedicated to each player. This implementation would have been more appropriate for CardsAR however it was more time efficient to implement hidden card's as they appear in the final build.

5.2.3 Device Feedback



CardsAR was designed to be a game that a player can remain seated for. Looking around at other players and leaning forward to select should be a natural part of the gameplay that a player feels comfortable doing. This appears to have been successful, but only with iPhone players. IPad players reported discomfort holding their devices, looking to select cards and pressing buttons at the same time. In hindsight CardsAR should have been limited to iPhones only.

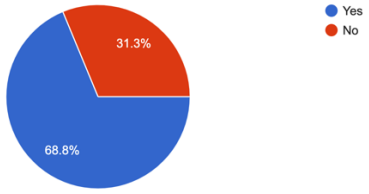
"...I felt the placement of the cards, and clarity on screen was good. There were some occasions I would have to move closer, but that was less to do with seeing the cards, and more to do with picking out the correct ones with the pointer."

"Big iPad, large spacing between the buttons and not so large hands made it quite hard sometimes especially when trying for fine movements like placing a pile of cards next to another"

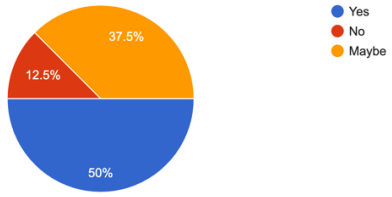
"A little due to size of iPad. Had to shift hand position a few times for comfort."

5.2.4 Social Feedback

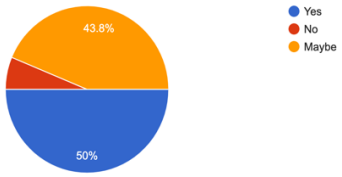
When other players were talking did you point your phone towards their avatars.
16 responses



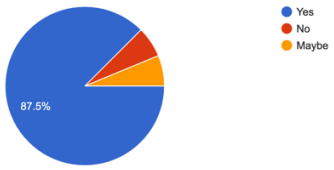
If the game included an optional video call feature, would you be inclined to use it.
16 responses



Do you think you would use an AR game like this to socialise remotely with friends right now
16 responses



Do you think you would have been more inclined to play the game during lockdown?
16 responses



The feedback on the social aspect of CardsAR was broadly positive. Players enjoyed the novelty of looking at each other's avatars even if the video call component wasn't present. Many showed interest in a version of this CardsAR that included video calling, and most showed an increased interest within the context of a lockdown.

"I really enjoyed seeing the picture frame move as people were talking. It added to the experience."

"It felt like a natural thing to do, because the avatars move as well it feels like a real conversation."

"It would be interesting to see how this worked with friends, particularly for the novelty. I imagine this would have been fun in lockdown!"

"I think I would have spent quite a lot of time on an app like this during lockdown, instead of something like Houseparty"

5.2.5 Feedback Summary

There are limitations to the formal testing that was undertaken. A group of 16 players is not large enough to draw definitive conclusions against. All members of the testing group personally know the author, which could have influenced their feedback. If CardsAR were a game from an anonymous developer, they might have been more critical and less interested in what it was trying to achieve.

However their responses provide enough data to partially answer some of the research questions. CardsAR succeed in creating an enjoyable experience for its players despite its poor multiplayer networking and awkward UI design. Testers have shown interest in playing a more refined version of this game. The core concept of CardsAR as an VR smartphone card game that utilises AR technology is viable. It would just require a significant redesign of the user interface in order to be more useable.

Chapter 6 – Evaluation Reflections and Conclusions

6.1 Project Planning

Core to the of design of CardsAR was the development of game that used multiple third party tools in ways that they had not originally been designed for. Difficultly encountered with these tools during development meant that compromises had to made to CardsAR. In particular The multiplayer netcode had to be redesigned midway through the final build and video calling needed to be omitted from the final build because of third party tool incompatibility and time constraints.

The problems encountered with Lightship ARDK were down to insufficient research. If more time had been spend reading the documentation during research, the limitations of Lightship would have been known before development started.

Leaving the multiplayer to build three was also a mistake. Build one should have been dedicated to establishing a demo of all the core components of CardsAR working in unison together. If the video calling limitations had been discovered in build one, more time could have been spend writing new iOS libraries to extend the Agora AI.

It could also be said that the scope for CardsAR was too wide. The card UI design was only partially successful, and the complete AR social video calling proved too difficult to complete within the time allowed. The project could either have been focused on designing an AR card game interface, or on socialising in AR. Merging both goals has meant that not enough time has been spent on either of them. The card playing interface in AR could have been a single player game that is focused on a more robust interface design. The social video calling in AR could have included a much simpler game, such as draughts or chess. This would have allowed for more time to be spent on the video call API.

6.2 Augmented Reality Tools

One of the design goals for CardsAR was to use the multiple input vectors that smartphones contain to provide a more fully featured AR experience. Contrary to dedicated handheld gaming devices, there is a broad diversity in smartphone software and hardwear in the marketplace. The apple iPhone ecosystem was the used for development and testing to help restrict the number of devices that might be used by testers. Even testing was limited to a smaller supset of the smartphone marketplace, the tools used for AR are targeted at the broadest possible market. This meant that the tools used in cards AR were designed around a lowest common specification.

Both Lightship ARDK and Manomotion used only one of the world facing camera lenses. The testing iPhone mini has a secondary wide angle lens that would have been particularly useful for the Manomotion hand tracking feature. The reasoning behind this decision not use the secondary lens is understandable, the wide angle lens shows more of the world but it also distorts image and makes closer items look further away. Switching between lenses would also change the rending perspective which could break the intended AR illusion. In addition to this, secondary lenses are not standardised across devices. Some have a secondary telephoto lens instead of wide angle,

some phones contain both. Supporting each type of lens as well supporting the transitions between lenses would be a great deal of work for perhaps not a lot of return.

[PICTURES OF WIDE ANGLE LENS]

In addition to diversity of hardware between manufacturers and ecosystems. Third party tools need to maintain support for older devices. When attempting the video call integration it was assumed that this would be possible as apple made it added multi camera capture to iOS in 2019 feeds (Apple, 2019). The Agora developers had not taken advantage of this feature yet so their plugin blocked AR functionality. It is possible that this feature was not a priority for as it required an iPhone with an a12 processor or newer. The a12 processor was released in 2018 but a significant portion of Agora may be using older models. At least one of the final build testers was using an a11 processor.

CardsAR has been designed to be a low-cost widely useable alternative to VR. The newest and most expensive iPhone Pro and Pro Max models provide LIDAR scanners for very accurate environment scanning. LIDAR provides a far more technically accurate AR experience, but LIDAR was not included in the design for CardsAR as limiting the game to those devices only was not low cost or widely playable. It appears that the same cost/benefit judgement was made by other third-party AR plugin developers. Only one camera lens in used in AR and multiple camera feeds is not possible on all devices so it has not been a priority for support.

6.3 Game Design

As covered in chapter 5 the current design of CardsAR was only partially successful. The UI could be understood after explanation but often the wrong buttons were pressed when looking at cards that the user didn't intend to, which lead to unexpected behaviour.

If the project were to be started again, the highlight mechanic would be replaced with a focus mechanic that allowed players to highlight cards only when they intended to do so.

Including text in the UI design could have improved readability as well. Doing so would have required a complete resign on the buttons as there is currently not enough room for text on each button. One possible option would be to have buttons that trigger drop down menus with text and icons.

6.4 Future Development

In addition to the changes to the UI above CardsAR has significant room for further development. The following features would all be required if CardsAR were to be a viable commercial product.

- The current multiplayer netcode should be removed and replaced with a game object based solution like Unity's netcode for game objects. Thiswould be more robust than the current chatroom API design.
- All card actions should be animated.
- The hidden card feature should be removed and replaced with a player hand mechanic like that seen in TableTop Simulator.
- A shared notepad would be a useful way for players to keep track of scores.

- Video calling should be possible with AR, so time needs to spent either extended the Agora API with a new iOS camera feed wrapper.

On top of the gameplay features listed above the current build has all session management and authentication hardcoded. A session manager interface would be required. Players should be able to create and join and invite friends to sessions.

If CardsAR were to be developed into a commercial product, it would incur ongoing costs related to data hosting and video call bandwidth. The game would therefore need to be sold at a cost or supported with in-game ads. In game advertising could even be rendered on a TV screen behind players that played ads at set intervals. This could also provide an interesting research project on how to integrate mobile ads seamlessly into a virtual world.

6.5 Personal Development

The author learned a great deal during the development of CardsAR. The difficulties encountered during development proved a valuable lesson prototype design. This project brought together multiple technologies which created development challenges that could have been identified early if the first build was better constructed. From the successes of CardsAR the author learned a great deal about the challenges of multiplayer game design and the architecture of a augmented reality experience.

6.6 Answers to Research Questions

1. Is it possible to build a tabletop card game using smartphone AR tools.
    a. CardsAR can be considered a working protype that does allow players to play cards in a virtual world using AR technology. The interface has significant room for improvement but the basic functionality works.
2. What limitations are there to user interface design when creating an AR virtual world game.
    a. When using a smartphone as a pointer device, the cards that players point towards need to be large enough to aim at. However larger cards lead to a more crowded gameplay area. This is a significant challenge that would require thorough user testing to find the right balance between size and space.
3. To what extent can smartphone AR technology be used to allow players to socially interact in a virtual world.
    a. The most popular 3rd party tools in unity for video calling do not allow for AR to be used at the same time. A workaround might possible but time didn't not allow for it within the constraints of this project. Other smartphone features like multiple camera lenses are not supported in AR tools which significantly reduces the feature set that is available to AR developers. Third party tools for AR are developed with the lowest common set of a hardware features in mind, therefore the toolsets will always be years behind the hardware improvements.
4. Is this method of socialising online enjoyable to casual game players?
    a. Testers were broadly disinterested in video calling while playing games. This changed during lockdown when more video calling took place. But that did not continue post lockdown. Some players showed interest in trying out a form of

CardsAR that included video calling but since that is not the game they played, their reaction to video calling in AR cannot be known at this point.

Glossary

References

Agora.io, 2022. *Agora Video SDK for Unity.* [Online]
Available at: https://assetstore.unity.com/packages/tools/video/agora-video-sdk-for-unity-134502
[Accessed 2022 December 2022].

Agora.io, 2022. *Token generators.* [Online]
Available at: https://docs.agora.io/en/video-calling/develop/integrate-token-generation?platform=android
[Accessed 19 December 2022].

Apple, 2019. *AVMultiCamPiP: Capturing from Multiple Cameras.* [Online]
Available at: https://developer.apple.com/documentation/avfoundation/capture_setup/avmulticampip_capturing_from_multiple_cameras
[Accessed 19 December 2022].

Apple, 2020. *Apple unveils new iPad Pro with breakthrough LiDAR Scanner and brings trackpad support to iPadOS.* [Online]
Available at: https://www.apple.com/uk/newsroom/2020/03/apple-unveils-new-ipad-pro-with-lidar-scanner-and-trackpad-support-in-ipados/
[Accessed 19 December 2022].

Apple, 2022. *Beta Testing Made Simple with TestFlight.* [Online]
Available at: https://developer.apple.com/testflight/
[Accessed 19 December 2022].

bankmycell.com, 2022. *HOW MANY SMARTPHONES ARE IN THE WORLD? (Source: https://www.bankmycell.com/blog/how-many-phones-are-in-the-world).* [Online]
Available at: https://www.bankmycell.com/blog/how-many-phones-are-in-the-world
[Accessed 2022 December 2022].

Chamary, J., 2018. *Why 'Pokémon GO' Is The World's Most Important Game.* [Online]
Available at: https://www.forbes.com/sites/jvchamary/2018/02/10/pokemon-go-science-health-benefits/?sh=6ac8b2053ab0
[Accessed 19 December 2022].

Climax Studios, 2017. *ARise - The Augmented Reality Puzzle Game.* [Online]
Available at: https://www.climaxstudios.com/portfolio-items/arise/
[Accessed 19 December 2022].

DAVENPORT, C., 2022. *Angry Birds returns to iOS and Android with no in-app purchases.* [Online]
Available at: https://www.xda-developers.com/rovio-classics-angry-birds/
[Accessed 19 December 2022].

Digite, 2022. *What is Kanban?.* [Online]
Available at: https://www.digite.com/kanban/what-is-kanban/
[Accessed 2022 December 2022].

Facebook, 2017. *Introducing face filters and more on Instagram.* [Online]
Available at: https://en-gb.facebook.com/business/news/instagram/introducing-face-filters?ref=search_new_1278
[Accessed 19 December 2022].

FRANCIS, C., 2016. *Unity, the Game Engine Behind Pokemon Go.* [Online]
Available at: https://rocketbuild.com/unity-the-game-engine-behind-pokemon-go/
[Accessed 19 December 2022].

Free Range, 2022. *Jenga AR.* [Online]
Available at: https://apps.apple.com/us/app/jenga-ar/id1547477000
[Accessed 19 December 2022].

Gambling Commission, 2021. *Exploring the impact of Covid-19 on gambling behaviour.* [Online]
Available at: https://www.gamblingcommission.gov.uk/statistics-and-research/publication/exploring-the-impact-of-covid-19-on-gambling-behaviour
[Accessed 19 December 2022].

gamesindustry.biz, 2020. *What is happening with video game sales during coronavirus.* [Online]
Available at: https://www.gamesindustry.biz/what-is-happening-with-video-game-sales-during-coronavirus
[Accessed 19 December 2020].

Garreau, J., 1994. *OFFICE MINEFIELD.* [Online]
Available at: https://www.washingtonpost.com/archive/lifestyle/1994/03/09/office-minefield/3b74132a-5f0a-455f-a04e-6171d023149b/
[Accessed 19 December 2022].

Gov.uk, 2020. *Prime Minister's statement on coronavirus (COVID-19): 23 March 2020.* [Online]
Available at: https://www.gov.uk/government/speeches/pm-address-to-the-nation-on-coronavirus-23-march-2020
[Accessed 19 December 2022].

Griffin, A., 2021. *Houseparty app shut down after keeping friends and families together during lockdowns.* [Online]
Available at: https://www.independent.co.uk/tech/houseparty-app-video-call-epic-fortnite-b1917654.html
[Accessed 19 December 2022].

IGN.com, 2021. *Among Us Wiki Guide.* [Online]
Available at: https://www.ign.com/wikis/among-us/How_To_Play
[Accessed 19 December 2022].

Interactive Design Foundation, 2020. *Augmented Reality – The Past, The Present and The Future.* [Online]
Available at: https://www.interaction-design.org/literature/article/augmented-reality-the-past-the-

present-and-the-future
[Accessed 19 December 2022].

Knezovic, A., 2022. *Coronavirus and Mobile Gaming: How did Covid-19 Impact the Mobile Game Industry?.* [Online]
Available at: https://www.blog.udonis.co/mobile-marketing/mobile-games/coronavirus-mobile-gaming
[Accessed 19 December 2022].

kunabi brother, 2017. *Euclidean Lands AR Trailer.* [Online]
Available at: https://www.youtube.com/watch?v=f6vA1s97_-0
[Accessed 19 December 2022].

Law, K., 2020. *11 (Not Obvious) Board Games You Can Play Over Zoom.* [Online]
Available at: https://www.vulture.com/article/best-board-games-zoom-quarantine.html
[Accessed 19 December 2022].

LEGO Systems, 2022. *LEGO® HIDDEN SIDE.* [Online]
Available at:
https://play.google.com/store/apps/details?id=com.lego.ar.hidden&hl=en_GB&gl=US
[Accessed 19 December 2022].

Manomotion, 2021. *Manomotion Community Projects.* [Online]
Available at: https://www.youtube.com/watch?v=ElJARRVnW-4&list=PLzNQkVGtWV6R_hFXX0ulJdtSwrq223HaW

ManoMotion, 2021. *Ring Try-On Tutorial.* [Online]
Available at: https://www.manomotion.com/ring-try-on-tutorial/
[Accessed 19 December 2022].

ManoMotion, 2022. *ManoMotion Mobile AR.* [Online]
Available at: https://www.manomotion.com/mobile-ar/
[Accessed 19 December 2022].

Meta, 2021. *Facebook Reports Third Quarter 2021 Results.* [Online]
Available at: https://investor.fb.com/investor-news/press-release-details/2021/Facebook-Reports-Third-Quarter-2021-Results/default.aspx
[Accessed 2022 December 2022].

Meta, 2021. *The Metaverse and How We'll Build It Together.* [Online]
Available at: https://www.youtube.com/watch?v=Uvufun6xer8&t=111s
[Accessed 2021 December 2022].

Newsweek, 2005. *GOING ALL IN FOR ONLINE POKER.* [Online]
Available at: https://www.newsweek.com/going-all-online-poker-117991
[Accessed 19 December 2022].

Niantic, 2022. *Lightship ARDK.* [Online]
Available at: https://lightship.dev/products/ardk/
[Accessed 19 December 2022].

Niantic, 2022. *Lightship ARDK Showcase.* [Online]
Available at: https://lightship.dev/showcase/

Niantic, 2022. *Sharing Virtual Object Information.* [Online]
Available at:
https://lightship.dev/docs/ardk/multiplayer/sharing_virtual_object_information.html#
[Accessed 19 December 2022].

Niantic, 2022. *Tutorials.* [Online]
Available at: https://lightship.dev/docs/ardk/tutorials.html
[Accessed 19 December 2022].

Noclip, 2022. *The Making of Among Us - Documentary.* [Online]
Available at: https://www.youtube.com/watch?v=IEBtB-WXaIY
[Accessed 19 December 2022].

Pagat, 2021. *Shithead.* [Online]
Available at: https://www.pagat.com/beating/shithead.html
[Accessed 19 December 2022].

Perez, S., 2016. *Pokémon Go becomes the fastest game to ever hit $500 million in revenue.* [Online]
Available at: https://techcrunch.com/2016/09/08/pokemon-go-becomes-the-fastest-game-to-ever-hit-500-million-in-revenue/?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAAGJaUAKhIZL4vZdrJdJXKvWOLHzYhI1kkoUZisvCoqYL_r977F37FydGH7Dr_ydJZA9C
[Accessed 19 December 2022].

Rutegard, J., 2021. *ManoMotion Hand Tracking SDK PRO For Unity 1.4.9 Is Now Available. Read More On What Is New !.* [Online]
Available at: https://www.manomotion.com/manomotion-hand-tracking-sdk-pro-for-unity-1-4-9-is-now-available/
[Accessed 19 December 2022].

Snapchat, 2022. *How do I use Lenses on Snapchat?.* [Online]
Available at: https://support.snapchat.com/en-US/a/face-world-lenses
[Accessed 19 December 2022].

Stuart, K., 2022. *Among Us: the ultimate party game of the paranoid Covid era.* [Online]
Available at: https://www.theguardian.com/games/2020/sep/29/among-us-the-ultimate-party-game-of-the-covid-era
[Accessed 2022 December 2022].

Sung, D., 2012. *Augmented Reality 2012: Is the dream any closer?.* [Online]
Available at: https://www.pocket-lint.com/ar-vr/news/qualcomm/116367-augmented-reality-2012-developments-glasses
[Accessed 19 December 2022].

System Requirtements Lab, 2022. *Tabletop Simulator System Requirements.* [Online]
Available at: https://www.systemrequirementslab.com/cyri/requirements/tabletop-simulator/12090
[Accessed 19 December 2022].

Tabletop Simulator, 2022. *Tabletop Simulator.* [Online]
Available at: https://www.tabletopsimulator.com/
[Accessed 19 December 2022].

The Game Kitchen, 2022. *All On Board! VR platform for board games.* [Online]
Available at: https://www.kickstarter.com/projects/thegamekitchen/all-on-board
[Accessed 19 December 2022].

The International Playing Cards society, 2007. *French cards.* [Online]
Available at: https://i-p-c-s.org/faq/history_6.php
[Accessed 2022 December 2022].

The Strong Museum of Play, 2019. *Microsoft Solitaire.* [Online]
Available at: https://www.museumofplay.org/games/microsoft-solitaire/
[Accessed 19 December 2022].

The Thing. 1982. [Film] Directed by John Carpenter. United States: The Turman-Foster Company.

TikTok, 2022. *Effects.* [Online]
Available at: https://support.tiktok.com/en/using-tiktok/creating-videos/effects
[Accessed 19 December 2022].

Unity, 2020. *How to choose the right netcode for your game.* [Online]
Available at: https://blog.unity.com/technology/choosing-the-right-netcode-for-your-game
[Accessed 2022 December 2022].

Warby Parker, 2022. *Warby Parker.* [Online]
Available at: https://www.warbyparker.com/app
[Accessed 19 December 2022].

Warren, T., 2020. *Microsoft Solitaire turns 30 years old today and still has 35 million monthly players.* [Online]
Available at: https://www.theverge.com/2020/5/22/21266718/microsoft-solitaire-30-years-old-history-birthday-record-attempt
[Accessed 19 December 2022].

Wearable Computer Lab., 2006. *ARQuake: Interactive Outdoor Augmented Reality Collaboration System.* [Online]
Available at: http://www.tinmith.net/arquake/
[Accessed 19 December 2022].

Yuan, E. S., 2020. *A Message to Our Users.* [Online]
Available at: https://blog.zoom.us/a-message-to-our-users/
[Accessed 19 December 2022].

Appendices

Appendice A: Project Proposal

Appendice B: Correspondence

B.1: Niantic

B.2: Agora

B.3: PubNub

Appendice B: CardsAR Code