AppledoreVR

Memory Palace in Virtual Reality

Adam Hynek

Aviral Garg

Emre Pekel

David Wong

Marinah Zhao

16.04.2017

Computer Engineering Design Studio II

TABLE OF CONTENTS

INTRODUCTION

From learning new languages to memorizing the periodic table of elements, the memory palace technique is a widely used and popular method of recalling information. It usually involves imagining a familiar environment with additional props, people, or other objects (mnemonics) added to help with recalling a specific memory. This technique is effective because users are able to use visualizations with spatial memory to quickly and efficiently recall information. Current memory palace use is limited to creating cue cards, drawing, or simply walking through a recognizable room or a common street. Our target market are students wishing to learn new languages and clinical uses to improve memory.

AppledoreVR is a platform that offers mind-mapping tools in Virtual Reality which create an immersive spatial environment, further engaging user visualization. Android users can take AppledoreVR anywhere with them and create custom memory palaces. Unlike traditional classrooms, AppledoreVR provides a combination of intuitive learning, innovation and a fun interactive experience.

The app uses a combination of Android features and the Unity game engine, along with external movement control from a DE2 board. Upon launching the app, the user can choose between various options, such as accessing a forum to discuss memory palace ideas, using their phone's camera to capture and save images of mnemonics that they can place in their memory palace, finding the last known location of their controller (DE2), and accessing their memory palace, which launches Unity.

Once in the virtual environment, they can choose from a list of objects and bring them into their memory palace, and are free to walk around using the external controller. The objects that the user spawns are stored so that the user can return to their palace at any time, even from a different device.

PROJECT MANAGEMENT

GITHUB

- **Code**

SCRUM

- **Biweekly meetings**

GITHUB ISSUES

- **Tasks:** milestones + labels + assignees
- **Bug reporting and fixing**

GITHUB WIKIS

- **Design Documents**
- **Troubleshooting documents**

Our project began as a single idea to create a memory palace in VR. As a team, we elaborated on features such as a forum, saving memory state, multiuser, etc. These features were decided very early in our project, so that we could have more time to implement them. There were two major Sprints to accomplish our final product. In the first sprint, we created a prototype of the essential features needed to make our product demoable. We also created deadlines for ourselves for features and exercises.

The project management decisions that contributed to our success were:

- The memory palace idea in Virtual Reality is an original idea that was developed by talking to Dr. Gary Lazar, psychiatrist with specialization in mood disorders and shares our interest and enthusiasm in improving memory techniques.
- We went out of our way to reach out to our prospective clients who would actually use our product. Following the principles of Agile Development, we used Evolutionary prototyping technique to iteratively change our product to our customer's needs.
- We conducted several interviews with professor Misuzu Kazama who teaches many Japanese course at UBC, other enthusiastic students who are learning Hiragana and Kanji (Japanese) and tested our project and provided much needed feedback for our project.

- Professor Misuzu also showed enthusiasm towards her interest in using our product in her classroom which boosted our confidence during the project and also, validated our implementation of the project.
- We conducted a debrief session after Module 1
- Idea research and functional design was finished early on during the reading break
- We created specific and detailed tasks for entire project immediately after Module 1 on Github Issues
    - We chose the essential features to implement for Sprint 1, and created prototypes for later features
    - For example, we implemented push buttons in Sprint 1 and later replaced the controller with a joystick which required an Analog-to-Digital-Converter
- Exercises were completed early on so that more time could be spent on the project
- We left more time than in Module 1 for integration in anticipation of potential problems and integrated features soon as features were done.
- Setup/troubleshooting documents were made for Asset Management, Unity Setup, Android Setup to share with team
- Better parallelism of tasks and ensured all our code was modular and tested before integration
- We met once a week outside of lab times and more so closer to our Sprint Demos. In addition to the lab times, we consistently updated each other on our progress through scrum meetings and helped each other with roadblocks
- Individuals discussed feature implementation with team members to get feedback on improvements to implementation before actually implementing the feature
- Individual meetings (online/in-person) with team-mates to meet our individual goals and discuss possible improvements to the workflow

WORK ACCOMPLISHED

SOFTWARE

Android App

a. Splash Screen
b. User Interface
c. Bluetooth Connection
d. User Authentication
e. Camera Functionality to send photos to Firebase
f. Forum for users to learn and interact
g. Google Maps Tracking of Controller

Unity

a. 3D Virtual Reality Environment creation
b. Dynamic 3D menu to for mnemonics selection
c. Spawning of Game Objects
d. Menu Selection of Game Object
e. Saving Game State to Firebase
f. Multi-user functionality (multiple users can edit one palace in real time)
g. Retrieving images uploaded by Camera/Web App and rendering them as a Game Object
h. Dynamic upload of current location images to firebase storage
i. Background music for UX

Web App

a. Uploading images and screenshots to firebase

Chrome Extension

a. Allows you to take screenshots of mnemonics

Firebase

a. Google Vision API for content moderation
b. User current position display from firebase storage

SOFTWARE CHALLENGES

1. Mnemonics Menu - since each object is sized in relation to its parent, it has a default size, world size and a relative size, which can all interact with each other.
2. User Authentication - in order for Google Sign In to work on every computer, an android.jks keystore file had to be created. Implementing and integrating sign in took a lot of effort due the long process outlined in Google's developer site (e.g. SHA-1, google-services.json, adding keys and signatures in multiple platforms), as well as Unity dependencies.
3. Creating "Remove Game Objects" functionality and having it be synchronized across multiple users was not straightforward. In order to select an object, there were multiple methods such as using a Physics Raycaster, adding in an event trigger and event system, or using a GVRRecticlePointer and adding in a collider onto the object. Usually, this is done through the Unity Inspector, but since our Game Object was instantiated at runtime, there was a specific combination of steps and code that would allow you to select the object with only the "camera" (VR interface), which took time to discover. After successfully selecting the object, Firebase's UpdateHandlers such as ChildRemoved Handler appeared to update only the objects existing, not objects removed, and also interfered with the current ChildAddedHandler, as you did not want to add/remove repeat items.
4. Using CameraView library
5. The Android/Unity workflow is awkward, such that the Android app is built as a plugin, which is then included by Unity when packaging the final application. Since the dependencies used by the Android end do not get included when being built as a plugin, and Unity does not automatically include many of these when building, many ClassNotFound errors were had. Eventually, this was more-or-less resolved by using Google's Unity Jar Resolver (https://github.com/googlesamples/unity-jar-resolver) and adding the required dependencies there, but initially much time was spent on getting Android and Unity to play nice together.
6. First time Unity/Firebase/Android developers
7. UX for mnemonic placement in mem-palace
8. The received Bluetooth data would be broken up into multiple signals, so these had to be put back together when looking for commands
9. Working with AsyncTasks for accessing the Firebase Storage (more information about the functionality in technical details for interesting features section)

HARDWARE

1. Analog stick for the user to control their movement inside the virtual reality application.
2. Analog to Digital Converter (ADC) to translate analog voltages from the joystick into digital signals to be retrieved by the NIOS II system.
3. Touchscreen to show instructions for the movement controls for the analog stick.
4. Bluetooth connection to send the digital signals from the ADC to the user's mobile device, as well as receive the user's id upon initial connection
5. GPS module to track the location of the controlling device.
6. Wifi connection to send the GPS coordinates to a Firebase database using the user ID received through Bluetooth

HARDWARE CHALLENGES

1. The drivers provided by Altera to interface with their SPI core were a little buggy, since they claim to be able to chain multiple commands together without revoking the chip select signal, but in reality, the chip select was being very briefly brought high between bytes. This issue was difficult to isolate, but was eventually resolved by sending multiple bytes as part of one driver call, which was also supported.
2. Another issue was that the ADC expects to be able to send and receive data at the same time, whereas the driver functions can only send *or* receive one byte at a time, but luckily it was still possible to alternate between sending and receiving bytes by shifting the commands without losing data.
3. The signal output by the ADC was also very noisy from the SPI clock and input signals that were next to it in the breadboard. This was eventually mostly resolved by placing a pull-down resistor (5.1kΩ) at the output.

DETAILED DESIGN

The NIOS II system inside the DE2 board is mainly responsible for delivering the digital data from the analog sticks to Android and the GPS coordinates to the Firebase database. The DE2 communicates with the mobile Android device using Bluetooth and with Firebase using Wifi.

How information was sent between the Android and DE2:

Movement commands sent to the Android device in the form of several different signals:

- A single character command for each of 3 buttons on the DE2: ‘L’, ‘F’, and ‘R’, as well as ‘b’ for clicking the analog stick.
- A 5-character command for analog stick readings, formatted like this: x**** and y****, for x and y directions, where **** is a 4-character number ranging from 0000 - 1023 (as we used a 10-bit ADC).
- On the Android end, the received Bluetooth data would be fragmented for commands larger than 1 character, so in order to detect commands, the past few data chunks would be stored, and concatenated with the latest received data. This was then searched backwards for valid commands (to get the latest ones first), and valid commands would update the stored values that get read every frame by the Unity game to dictate a corresponding action.

FLOWCHART OF SOFTWARE

Figure 1. High Level Design Diagram

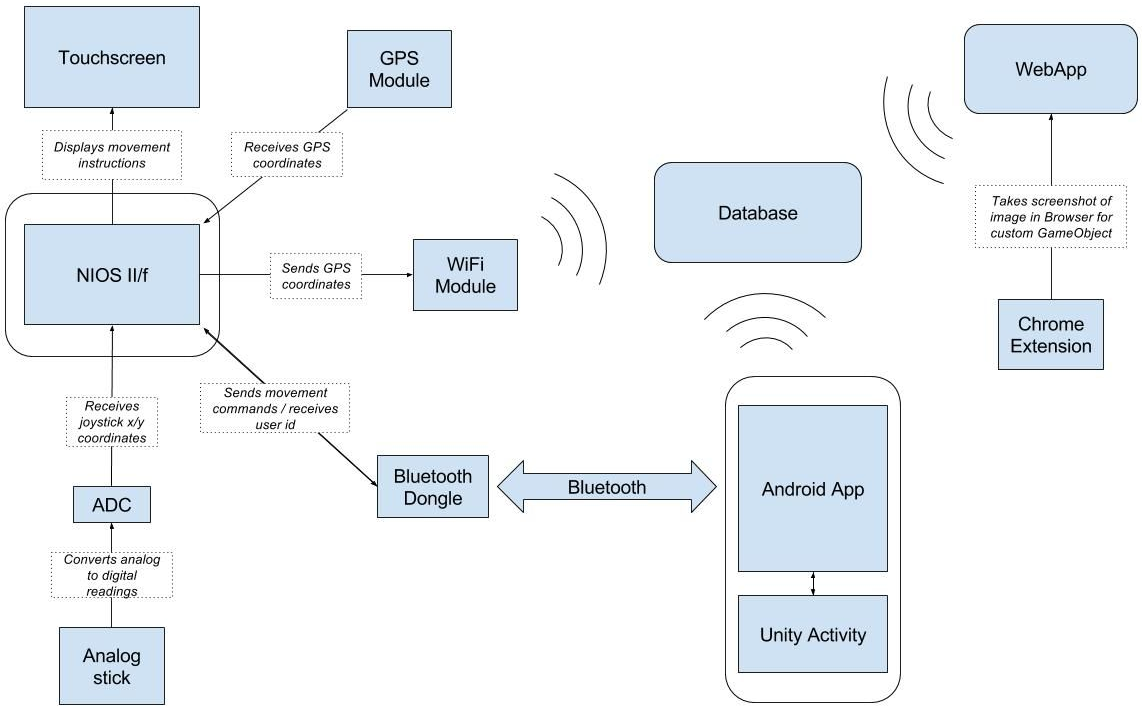


Figure 2. Android Software Diagram

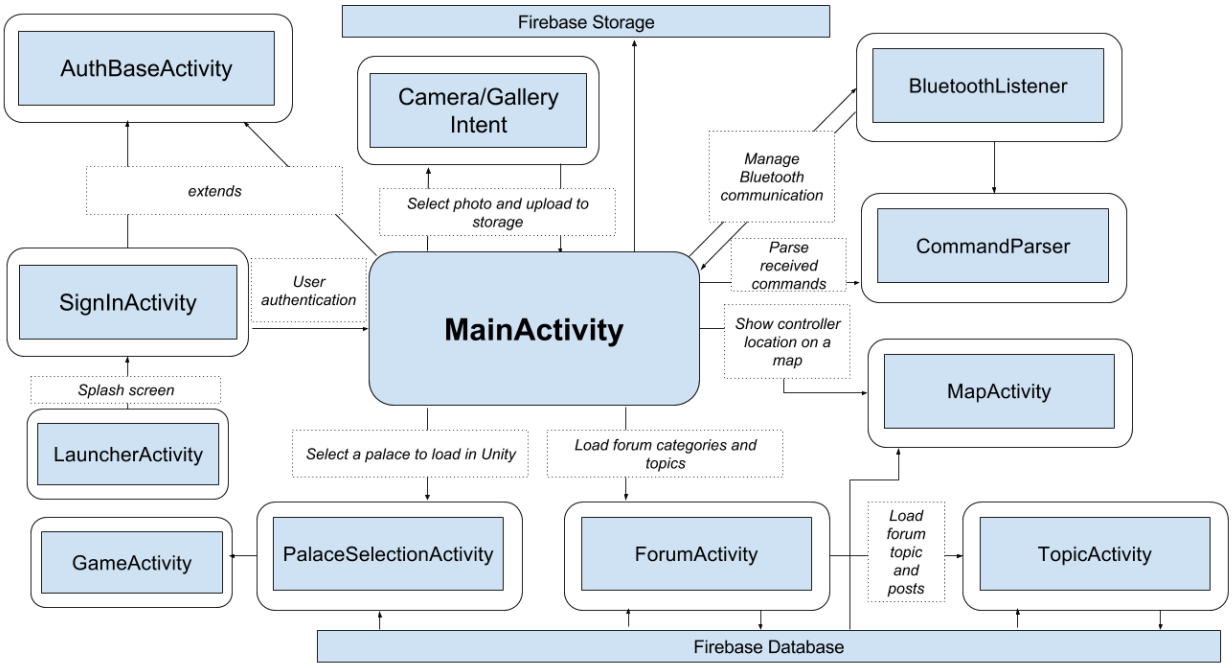
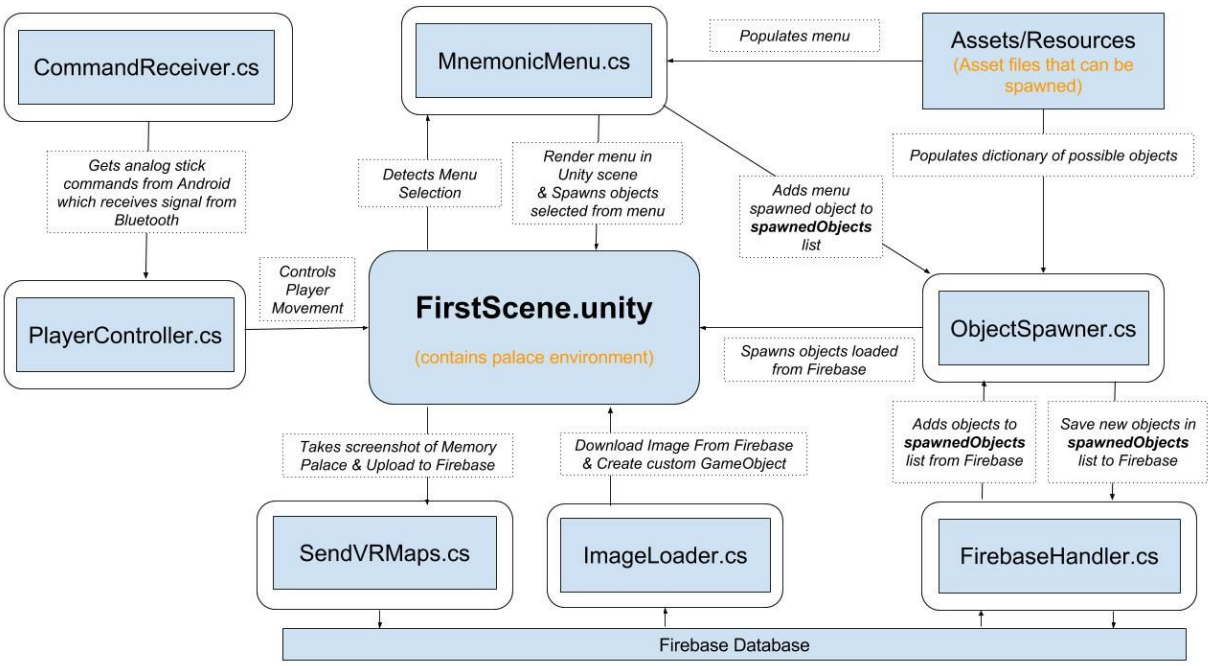


Figure 3. Unity Software Diagram



MAJOR DATA STRUCTURES

DATA	FUNCTION	DETAILS
ForumCategory	Data class for forum categories	Fields: *text, description;* Used for listing forum categories
ForumTopic	Data class for forum topics	Fields: *topicName, topicId, category, posts;* Used for storing topic information in TopicActivity
ForumPost	Data class for forum posts	Fields: *postId, userId, userName, message, avatarUrl, timestamp, category, topicId;* Used for storing forum posts for a certain topic in TopicActivity

ListForumTopic	Data class for forum topics to be listed	Fields: *title, timestamp, topicId, userName, category, replies, views;* A different class from ForumTopic was required for listing, due to the number of replies and views
User	Data class for users	Fields: *name, email, uid;* Stores user information and contains the function to upload a user to the database on initial sign in
GPSCoord	Stores GPS coordinates	Fields: *latitude, longitude;* Stores GPS data received from Firebase used in MapActivity
UniqueGameObject	Data class for game objects	Fields: *gameObject, uid, palaceUserID, typeName, position, rotation* Contains structure of game object to save/load the game state to/from Firebase

TABLE OF SOFTWARE CONTENT

ACTIVITIES	FUNCTION	DETAILS
LauncherActivity	Splash screen	The first activity launched on start of the application
AuthBaseActivity	Contains base authentication functions	Superclass for both SignInActivity and MainActivity
SignInActivity	Sign in screen and user authentication with Google Sign-In	Intent from LauncherActivity Immediately launches MainActivity if the user has signed in before. On one unsuccessful sign in attempt, user can choose to continue anonymously.

MainActivity	Main UI, the center point of navigation in the app	Delegates button presses for starting Unity, forum, controller locator, and camera. Initializes and manages Bluetooth connection, including sending the user id to DE2. Handles uploading a photo to Firebase Storage. Prompts for all the application permissions needed.
ForumActivity	Retrieves and displays the topics from the database based on the category	Activity launched on click of the forum button in MainActivity
TopicActivity	Retrieves and displays the forum posts from the database based on the topic	Activity launched on click of a ListForumTopic item in ForumActivity
MapActivity	Retrieves the GPS coordinates of the controller from the database and displays them on the Google Map	Can navigate and adjust (zoom, move) the Google Map
PalaceSelectionActivity	Allows user to selects which user's palace they want to visit in Unity	Activity launched on click of the Unity button in MainActivity
GameActivity	Launches Unity	This activity extends a Unity activity that launches the packaged game when started
ADAPTERS	FUNCTION	DETAILS
CategoryAdapter	ListView adapter for forum categories	Displays the category and its description
TopicAdapter	ListView adapter for forum topics	Displays the topic name, original poster, timestamp, number of replies, and number of views
MessageAdapter	ListView adapter for forum posts	Displays the name and avatar of the poster, the message, and the timestamp

PalaceAdapter	ListView adapter for memory palaces	Displays the name and owner of the palace
ANDROID CLASSES	FUNCTION	DETAILS
BluetoothListener	Handles Bluetooth communication	Uses multi-threading for listening incoming connections, connecting to a device, and the connected state
CommandParser	Parses commands received through Bluetooth	Since Bluetooth data comes in fragmented, this stores some past data and looks through all of it for valid movement commands, and stores current received command values
FetchDataCallback	Simple interface for updating a ListView	Used for calling ArrayAdapter's notifyDataSetChanged() function after fetching data from the database
ForumEditText	Custom EditText view for the forum	Required in order to override onKeyPreIme() function to make the EditText lose focus on back press
UNITY CLASSES	FUNCTION	DETAILS
MnemonicsMenu	Allows selection of object to spawn	
ObjectSpawner	Handles game state and object spawning	Spawns game objects Stores all spawned objects Uploads "game state" (spawned objects) to Firebase by attaching uid, typename, etc Loads existing game state from Firebase
FirebaseHandler	Handles Firebase connection	Initializes Firebase connection Contains Firebase update code to get changing game state in real time
CommandReceiver	Transfers commands from Android to Unity	This class makes several calls through Unity's

		AndroidJavaClass API to the CommandParser class (above) to get the current movement commands received through Bluetooth, and affects the player controller appropriately
PlayerController	Controls the player	This is a modified version of Unity's included RigidbodyFirstPersonController class, which moves the player around, jumps, etc.
ImageLoader	Applies an internet image to a texture	This class downloads an image from a url and sets the texture of the object it's attached to to the downloaded image

HARDWARE MODULES

MODULE	FUNCTION
Analog stick	Tracks the desired movement from the user
Analog to Digital Converter	Converts the analog voltages from the joystick into digital data and sends them to the NIOS system
GPS	Detects the current location of the hardware device
Bluetooth	Delivers the ADC data to user's mobile device, and receives the user's id upon initial connection
Wifi	Delivers the GPS data to the Firebase server

TECHNICAL DETAILS OF INTERESTING FEATURES

On the Android side of our application, there is an option to choose a picture from the gallery or take a picture with the camera. The chosen picture is to be used as a mnemonic in the user's memory palace. Due to the Firebase Storage library not allowing easy retrieval of the list of files in the storage, a recursive algorithm was implemented. To elaborate, images from the storage are displayed on the Unity side by going through consecutively numbered filenames, and iterating over the storage files was necessary on the Android side to name the new image to be uploaded. Firebase Storage provides AsyncTasks for retrieving a URL from the storage based on the name, and uploading a file to the storage. These AsyncTasks can have onFailure and onSuccess listeners that allow specific behaviour for when the task is successful or not. In order to find the next available file name in the storage;

- A searchTask attempting to retrieve the URL of a given filename is created
- On success, meaning a URL was found, a global variable called mFileIndex is incremented to search for the next file name, and the function is called recursively
- On failure, meaning a URL was not found, the callback function that uploads the image to the storage with filename mFileIndex is called

Then, the image is uploaded asynchronously and the user is notified of the result with a Toast.

RESULTS

According to our Requirements & High Level Design document, we have accomplished all of our requirements and more. We were able to accomplish our requirements early and enhanced and added additional features into our project. TAs and team members enjoyed playing with the final interactive product.

ORIGINAL REQUIREMENT Users should be able to …	FEATURES IMPLEMENTED
Navigate through a "memory palace" they have created, in virtual reality	Created memory palace in Unity which saves your game state in real time and allows you to spawn memory objects through the palace
Visit other people's memory palaces	User authentication through Firebase and saving of game state in real time allows user to visit other people's memory palaces by listing other user's memory palaces in Palace Selection
Edit (add/remove) objects to their memory palace	Objects in memory palace can be added (remove has not been integrated)
Control movement through external controller (FPGA)	FPGA Pushbuttons allow users to make selections and "jump" while wearing the headset Analog Stick allows users to navigate through user palace and make selections
Talk with other people in the VR-One community	Android forum allows users share mnemonic ideas and learning techniques online

In order to test for robustness, our hardware has been well tested since Module 1. Newly added features such as the ADC and analog stick were also well-tested before integrating. Each team member pulled the latest code onto their own branch and integrated and tested the product on their computers before merging it into the main product. Some team members did not have an Android phone, so that made it difficult to test on the device; however, Unity emulator provided a good representation of the device.

INTEGRATION OF STANDARDS

For the Android features implemented, we followed the design standards to legitimize our app with a professional appearance. We also applied the three design principles to make our application user-friendly.

(***Note: Please refer to Appendix for Screenshots***)

CONFORMING TO STANDARDS

Android Design guidelines are followed to not confuse the user. The application adheres to the design standards by not redefining the expected function of system icons (i.e. Back button) and not replacing system existing system icons (*see Screenshot Appendix h*).

Conforming to *Accessibility* principles, all menu elements are organized and have sufficient black-and-white contrast with appropriate size. Our application is simplified by not cluttering the screen, and each element has a distinct call to action (*See Screenshot Appendix c*).

Following *Material Motion* principles, our transitions are quick to keep the user from waiting longer than necessary, and the motions we applied are simple and smooth.
(*Note: Please refer to the group video on Youtube to examine this part*)

Less commonly used features are hidden away from the main user screen, but remain easily accessible through a button in the top righthand corner.

DEVIATING FROM STANDARDS

Deviating from *Accessibility* and *Style* standards, the application uses a black-, and-white theme (without other colours). Adding more colours into the theme could provide more distinction to different elements, but the simplicity of our application allows the user to easily understand the functions without additional colours. However, the Memory Palace within the Unity software will provide the user with a colourful experience.

Imposing standards may hinder creativity, but standards allow designers to make technology safe and accessible to users, and allow different developers to modularize and share components knowing they follow the same standards. Imposing standards also prevents developers from creating new principles that may accomplish the same goals as the original standards and prevent code reuse. Design standards also create a monoculture of applications that look and feel similar which is uninteresting to developers and users.

CONCLUSION AND SUGGESTED FUTURE WORK

Summary

Incorporating Virtual Reality into a FPGA project was an ambitious decision since we were all novice to this field of technology. We could have modified and reused our Module 1 Project, but brainstorming a new idea allowed us to take risks, learn new technologies, and create a project that was distinctive from the other groups.

We learned from Module 1 and greatly improved our team performance. Group members were more concentrated on the project and early integration ensured the product could be finished on-time and include plenty of interesting features. Each member was assigned important components, whilst keeping track of each other's work and group-made deadlines. Once a member completed and integrated a component or feature, the member take on the next important task.

In conclusion, the team was productive and efficient by sharing the same ambition to create a successful project. Not only did we complete all the features listed in our original Requirements & High Level Design document, but also added additional work-heavy features as a result of our communicative and goal-orientated group dynamic.

Improvements

For future work, AppledoreVR can be extended by improving our current features by having forum extensions such as searching, sorting, upvoting, editing, and deleting. We would like to allow each user to have multiple palaces per user (currently one palace per user). We can also integrated our "Remove game objects" functionality after testing. Lastly, we can add Bluetooth discoverability for pairing with the associated device at the first setup.

AppledoreVR would also like to add unique features such as having in game editor to create custom game objects during runtime.

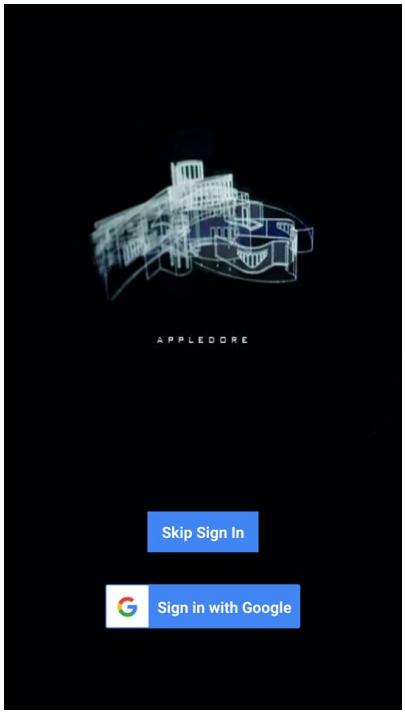
Overall, the next goal for AppledoreVR is to move from prototype stage to product stage. AppledoreVR can be extended to move away from Android and support HTC Vive or other more advanced VR Headsets so that a user's physical movement can be detected and controller extensions can be added, thus replacing the FPGA, ADC, and analog stick unit.

Firebase is made to scale which will allow AppledoreVR to support many users and the end goal would be to put AppledoreVR in the Google Play Store, Apple Store, and have VR-One be used in clinical/educational uses.

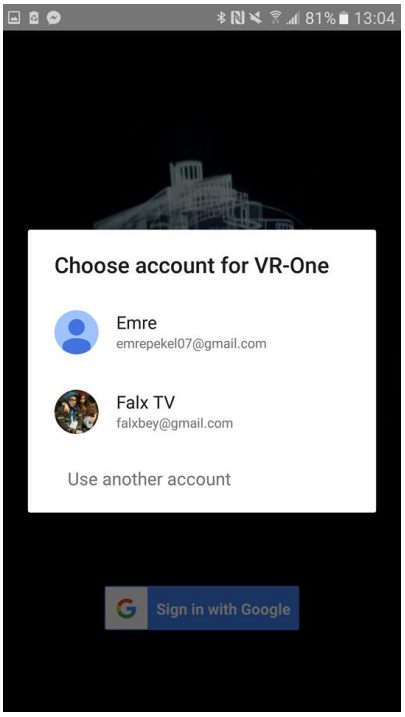
SOURCE CODE

https://github.com/AviralGarg1993/VR-One

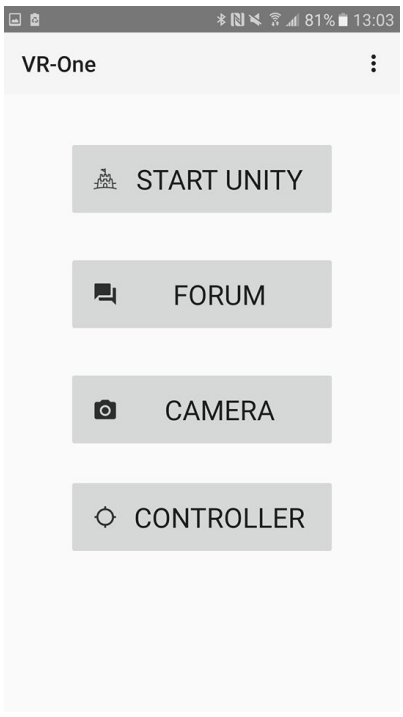
SCREENSHOTS APPENDIX



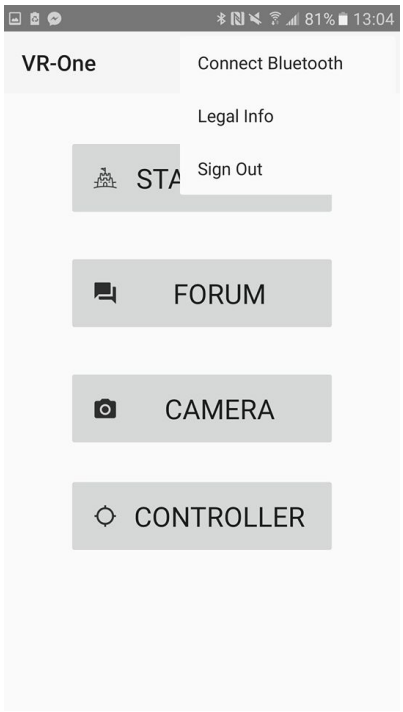
(a)



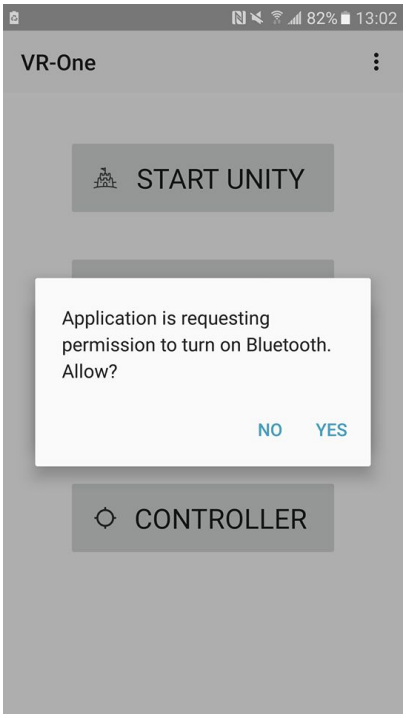
(b)



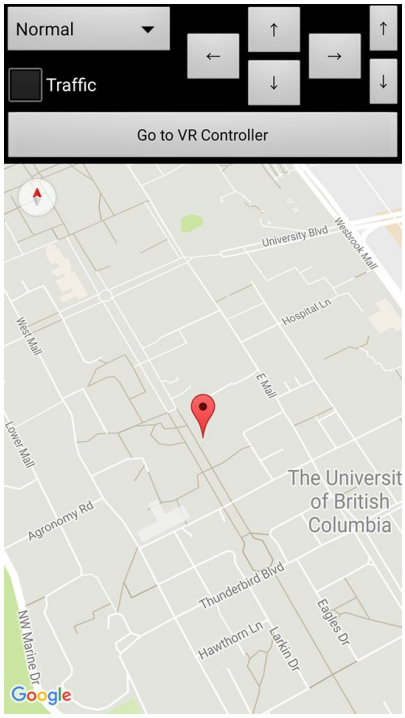
(c)



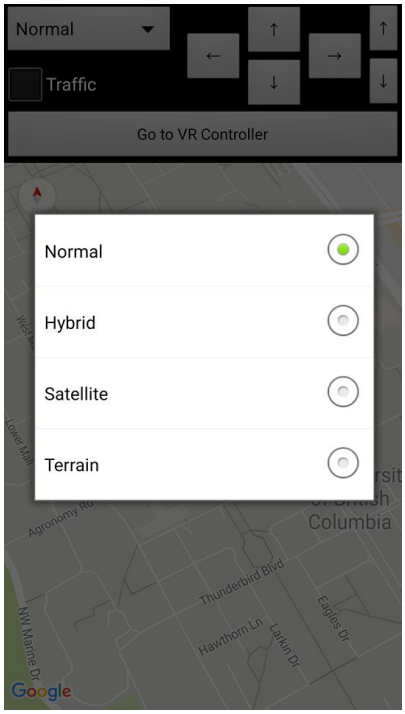
(d)



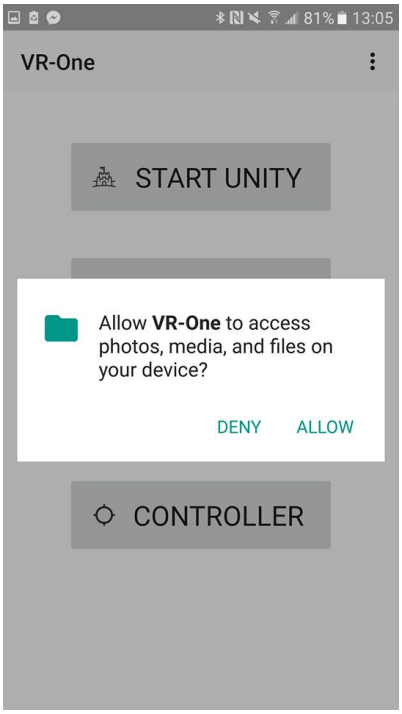
(e)



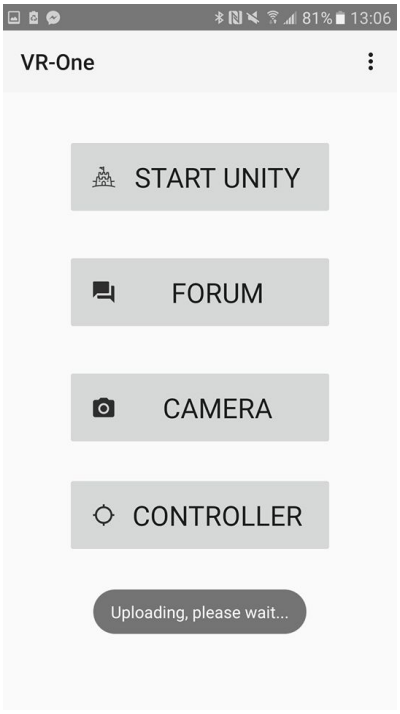
(f)


Normal
Traffic
Go to VR Controller
Normal
Hybrid
Satellite
Terrain
Columbia
Thunderbird Blvd
Hawthorn Ln
Larkin Dr
Eagles Dr
NW Maine Dr
Google

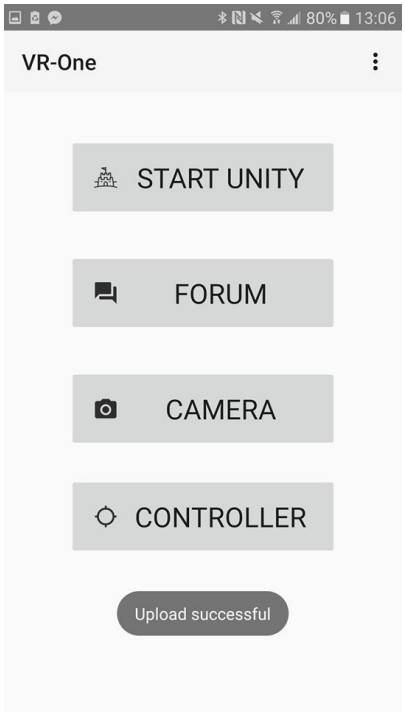
(g)


81% 13:05
VR-One
START UNITY
Allow VR-One to access photos, media, and files on your device?
DENY
ALLOW
CONTROLLER

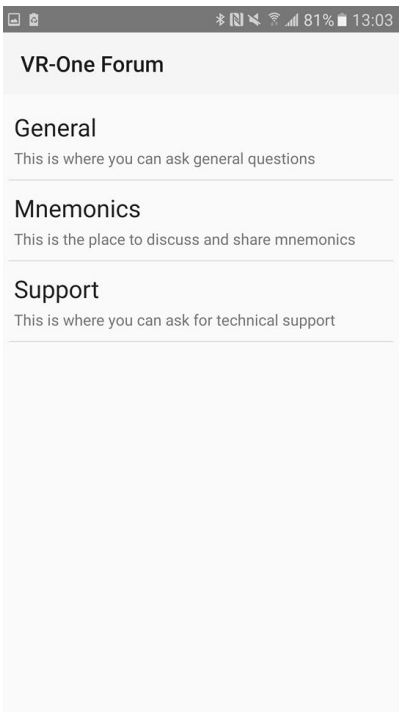
(h)


81% 13:06
VR-One
START UNITY
FORUM
CAMERA
CONTROLLER
Uploading, please wait...

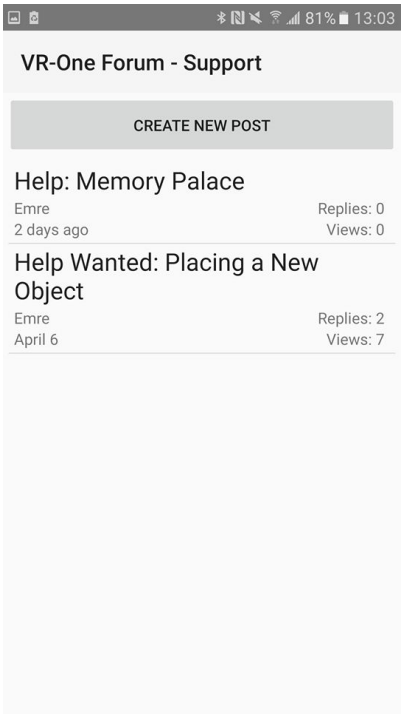
(i)


80% 13:06
VR-One
START UNITY
FORUM
CAMERA
CONTROLLER
Upload successful

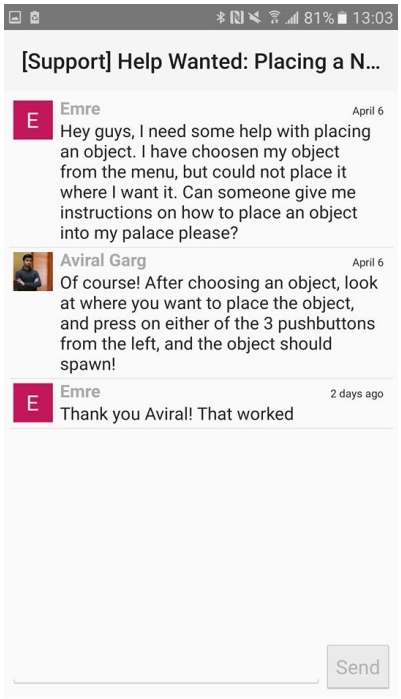
(j)


81% 13:03
VR-One Forum
General
This is where you can ask general questions
Mnemonics
This is the place to discuss and share mnemonics
Support
This is where you can ask for technical support

(k)


81% 13:03
VR-One Forum - Support
CREATE NEW POST
Help: Memory Palace
Emre
2 days ago
Replies: 0
Views: 0
Help Wanted: Placing a New Object
Emre
April 6
Replies: 2
Views: 7

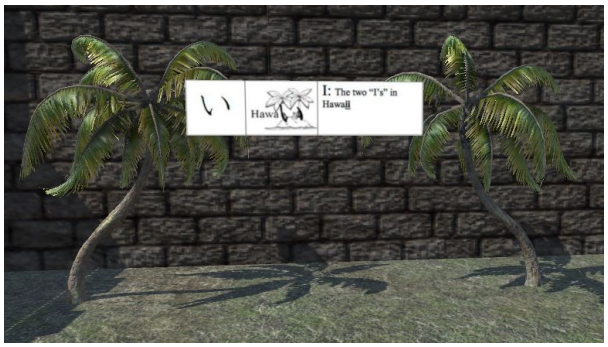
(l)


81% 13:03
[Support] Help Wanted: Placing a N...
Emre
April 6
Hey guys, I need some help with placing an object. I have choosen my object from the menu, but could not place it where I want it. Can someone give me instructions on how to place an object into my palace please?
Aviral Garg
April 6
Of course! After choosing an object, look at where you want to place the object, and press on either of the 3 pushbuttons from the left, and the object should spawn!
Emre
2 days ago
Thank you Aviral! That worked
Send

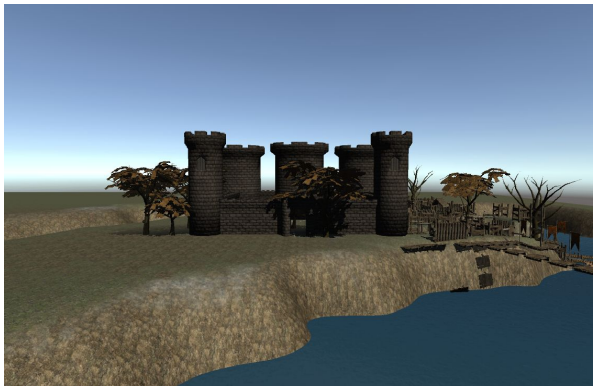
(m)


Categories
Animals
Castle
Food
Menu

(n)


I: The two "I's" in Hawaii

(o)



(p)

INDIVIDUAL APPENDIX

ADAM HYNEK

1. **Individual Contribution**
   Integration of Android Studio with Unity; adding hardware and software for SPI communication with the ADC; bluetooth communication between the DE2 and the Android app to relay movement input; communication between the Unity game and the Android app to access bluetooth commands stored in the Android app from Unity; downloading images in Unity and displaying them on objects; helped team members with Unity stuff; general design ideas
2. **Team Effectiveness / Other Comments**
   Since this module was larger in scope and had more distinct separated components (instead of everyone working on hardware/firmware, there was now a DE2 side, an Android side, and a Unity side, with some overlap), it was easier for people to work on different components without worrying too much about integrating with everyone else (a big problem in module 1).
   Overall, team effectiveness was much improved from module 1, but perhaps we should have stopped implementing new features a little earlier and focused more on fully polishing the existing ones.

AVIRAL GARG

1. **Contribution**
   Individual:
   Project idea, project management, project requirements engineering, Unity virtual reality environment, project tools research, exercises (2.1 and 2.3), Android camera feature, Adult content moderation using Google Vision API, Google Chrome extension, web app, initial multi-user chat version for Forum, logo design and more.

   With Team Members:
   Requirements engineering and design for Android forum, dynamic mnemonic image loading, Firebase database design, and more.

2. **Team Effectiveness and Other Comments**

   I finished 3 online courses on Virtual Reality and 1 course on Software Development Processes early on during the reading break. That allowed me to do almost entirety of project idea generation, requirements gathering,analysis and engineering for the project, project tools discovery (Firebase/Unity/Github issues/Github wikis), Virtual Reality environment, tasks creation, division and so much more in terms of project management, project design, client interviews and more in the first two-three weeks of the module 2, it allowed a substantial amount of time for the team to focus on their tasks. One major improvement our team could have made is instead of relying on code working on one person's machine, following the documents on Github wiki and have the entire project working on everyone's machine. That would have substantially improved parallel work.

DAVID WONG

1. **Individual Contribution**

   Adding Wifi port into the serial ports (x86 decoder) of FPGA for sprint 1; Adding Touchscreen UI for the final demo; Redesigned the NIOS system (top-level vhdl) of FPGA when adding SPI; Adding some graphics to Android UI (not integrated); Code refactoring for C files on the hardware side; Exercise 2.2

   With Team Members

   Adding hardware and software for SPI communication with the ADC (in conjunction with Adam Hynek); Providing suggestions to Adam in parsing ADC input on Android from Bluetooth and relay movement; Adding object-assets (mnemonics) to the Unity workspace (in conjunction with Marinah Zhao); helping other teammate to debug some errors on the software side

2. **Team Effectiveness & Other Comments**

   Comparing to module 1, AppledoreVR has greater successes in terms of group dynamic, group communication, and individual & group efficiency. The brave decision in making a Virtual Reality project with FPGA has led the team to plan out the development and design of the product carefully in the beginning stage. Following that, each member of the team would take components of the project relating to his/her strength, skills, and past module experiences. Despite all the members had no experiences with Unity programming and adding external analog device and ADC to the DE2 board, the team could overcome many difficulties and roadblocks through discussions and meetings, thus strengthen the group dynamic and effectiveness. Learning from previous module's experiences, the team would integrate the completed components/features without delays which increases the efficiency of project development. However, since the project itself is quite massive and complex, the team should restrict the numbers of features to be added later and focus more on improving the existing core functionalities of the product.

EMRE PEKEL

1. **Individual Contribution**
    a. Bluetooth communication, for both Android and DE2
        i. Android: Implemented multi-threaded Bluetooth listener
        ii. Android and Unity communication and integration (in conjunction with Adam)
        iii. Sending user id to DE2
    b. C code for sending pushbutton inputs through the dongle (Sprint 1)
    c. C code and LUA script for making an HTTP PATCH request to Firebase with the GPS coordinates of the controller (Sprint 1)
    d. Refactored C code for the DE2 side (Sprint 1)
    e. Touchscreen UI for the demo (Sprint 1)
    f. Google Sign-In
    g. Android Forum, including;
        i. Creating a new post / replying to an existing post
        ii. Real-time user interaction
        iii. AsyncTask for loading user avatars without blocking the UI
        iv. UI and minor necessities
    h. Refined Android UI, refactored Java code
    i. Camera/gallery functionality
        i. Implemented algorithm to find next available file name and upload image to Firebase Storage
    j. Debugged issues on C and Android side, as well as Unity dependencies
    k. Downloaded suitable Unity assets for memory palaces
    l. Wrote Java and C# code for sending user id and palace id from Android to Unity
    m. Brainstormed general design ideas and enhancements
2. **Team Effectiveness**

    Compared to Module 1, our team effectiveness was improved significantly. I believe the main factor for this was the better use of GitHub. We have created issues for tasks, bugs, and enhancements where everyone could see and pick from. Moreover, we usually met (as a team or in smaller groups) outside of the lab hours, which improved our communication. Due to working on different modules and tasks, and the nature of our projects, integration was smoother compared to Module 1. One possible improvement would be even better communication and

parallelism, so that every team member can focus on their tasks and easily run the project on their device; which could also lead to better workflow.

3. **Other Comments**
    a. I made sure my code was modular and followed the principles of Android programming and object-oriented design, where possible. Moreover, I tested my code with multiple devices in order to make sure it worked as expected. In addition, constantly communicating with my teammates helped integrate my tasks with the group.
    b. One of the biggest hurdles I have had to overcome was integrating Google Sign In with Unity, and making it work for every user. As outlined in the challenges section, it took a lot of time to successfully implement user authentication. Resolving dependencies, working with AsyncTasks (i.e. ensuring that some code portion is executed after the task is successful), making HTTP PATCH requests to Firebase (had to flash new custom firmware with tls module to solve certificate problems) were some of the other hurdles I have had to overcome.
    c. I have communicated with my teammates to make sure everything was going good, and helped them debug software problems. I took on most of the Android side of the project, and made sure to communicate my progress to the team. I exchanged ideas and offered my help to the team whenever possible.
    d. I have mainly improved my Android skills, as well as my general software design skills. I have learned to ask questions and facilitate brainstorming sessions for ideas to be enhanced. I have learned to take suggestions and implement them appropriately. I have learned to be better organized as a team, as well as an individual.
    e. There were not any major team dynamic issues. The tasks could have been allocated better and more fairly; however, this did not present a major problem.
    f. Apart from contributing to the documentation, I have done the production of the short video introducing our project.

MARINAH ZHAO

1. **Individual Contribution**
    a. Make hardware code compact and modular from Module 1 (with Emre)
    b. Wrote Push buttons code on FPGA
    c. Exercise 2.4 Google Maps Exercise
    d. Added Map Activity to Android
    e. Initial Screen UI on Android for Sprint 1
    f. Help debug Android to Unity dependencies for connection (with Adam)
    g. Created prototype of menu in Unity
    h. Implemented saving/loading Game State in real-time to Firebase
    i. Implemented multiuser (used Emre's palace ID)
    j. Looked for suitable Unity Assets and Mnemonics
2. **Team Effectiveness**
    a. Effectiveness of our team greatly improved in Module 2 compared to Module 1. Our project idea and tasks were formed early on and each teammate asked for tasks to do when they were finished their current task. One or a pair of team members accomplished almost all the features in parallel, which allowed us to create many features.

       The team also helped each other by explaining their own component so that others can help or quickly join in on the expanding the project. Getting feedback from teammates on how to implement a feature before implementing the feature saved the team time because possible pitfalls or better implementation ideas could be brought up.

       Documenting our design decisions together so that each team member knew which component they were working on allowed for a shared understanding of each other's components. This made integration easier in the end. Sharing of code was also greatly improved in Module 2 including documentation on setup. Communication was also constant in Module 2.

       Improvements could be integrating sooner (improved compared to Module 1) and making time to meet earlier so that team does not have to work till 3AM day prior to Demo.

3. **Other Comments**
    a. I tested my components before integrating with the group and requested to

have updated code to integrate onto.

b. Hurdles that I had to overcome were learning Unity for the first time since it contained physics-based features (Game/Visual Programming).
c. I helped ensure success by always contributing to the project. If I was finished my task, I looked for the next task by asking the team or coming up with my own (i.e. improving UI). If we had tasks that were bottlenecks, I helped debug them (i.e. Android to Unity connection). Overall, I listened and communicated with team which greatly reduced any undone or redundant work.
d. I learned to stand up for myself and ask questions to the team where I didn't understand. I learned to attempt to finish tasks earlier so that it can be discussed and integrated with the team, in case problems arose. I learned to seek and listen to team member's feedback and also give my own feedback where things can be improved.
e. There were no team dynamic issues other than distribution of the work. If a team member was lost when the project became large and complex, a learning curve developed which made it difficult for a team member to contribute, whilst the other team members had to deal with urgent software issues closer to the deadline. Overall, the team worked well together and I enjoyed my time on the team.
f. I spent a lot of time attempting to use interrupts and creating a prototype for the menu. Unfortunately, those were unsuccessful, but stopping and moving on to another task was more productive (since basic pushbuttons worked without interrupts) and Aviral was able to more easily create a menu. I then moved on to the task of adding "Remove functionality" which did not have time to be integrated.