2019-11-22 Biweekly Report

Overview

By the end of this week we have researched several aspects relating to this project, including target platform research and feature prioritization. However we have also came across several major problems while developing. We aim to resolve the problems by next week.

Tasks completed

1. Housekeeping.
    a. Project progress-tracking and documentation via Jira and Confluence.
    b. Cleaned Initial Project Brief.
    c. Added meeting minute (for 09/10/2019) for record.
2. Added Feature Prioritization via MoSCoW list.
3. Created user personas for attendee and presenter.
4. Completed related project research.
    a. Analyzed three major related existing products (with respect to VR Meeting Environments); compared these products' strong and weak points to assist in our project
        i. MeetinVR
        ii. Microsoft HoloLens 2
        iii. Doghead Simulations' rumii.
5. Decided on graphics engine (Unity).
6. Researched Oculus Go.
7. Researched real-time voice chat solutions.
8. Human Computer Interaction
    a. VR UI Sketches

Progress Evaluation

The project is running on time but a bit late compared to our initial estimation, because of some blocking issues that we need to resolve with respect to the target platform. We want to starting coding by around end of week 3, but it looks like we will have to defer it to early week 4, as we need to do some additional research and design.

Problems and Considerations

1. Voice chat solution.
    a. Achieving real-time voice chat remains to be a technical problem:
        i. Sender voice collection.
        ii. Transmission of voice across IP networks.
2. Difficulties when developing with Oculus Go.
    a. We find it very hard to develop for Oculus Go as a target platform, because of various aspects:
        i. Difficult to run the application on Oculus Go.
            1. With respect to Unity integration, we are restricted to use Android debug bridge (adb), self-signing a debug certificate, sending the build to a phone running Android, and using the Oculus companion app (and after pairing it with the Oculus Go headset) to send it to the headset. This is a very long deployment cycle, even for local development.
        ii. Documentation and examples lacking.
            1. Oculus Go development (with Unity integration) has some documentation, and very simple examples, but we find that it's difficult to search for information we need.
            2. In almost every aspect, documentation exists but is very brief. When we try to research further, we are unable to find much more information - it is very opaque.

iii. Oculus Go, as a standalone VR gear, practically has no support for file browsing, which makes any attempt for presentation file uploading very difficult.
iv. Oculus Platform SDK has VoIP [1] but it is very restricted:
    1. It heavily relies on Oculus provided servers and restricts users to use Oculus Rooms and Parties API. But based on reviews on [2], developers have encountered many, many problems, and is not supported anymore. Such information is not up to date on the developer documentations site.
    2. We are also unable to find suitable C# libraries supporting real time voice communication development.

Proposed Alternative Target Platform for the Prototype

We propose a laptop-based or desktop-based VR meeting experience for the prototype:

Benefits:

- Similar to a 3D laptop / desktop game.
- Laptop and desktop operating systems (e.g. Unix-like OSes such as macOS) are much more common and standardized.
- Users do not need an additional piece of hardware (i.e. VR head gear).
- Users have much more varied input devices (keyboard and mouse).
- We are then able to easily to networking, filesystem and other types of I/O.
- Development and deployment is significantly easier.
- More development documentation exists.
- More existing libraries supporting real time audio - we are also not necessarily restricted to only the C# ecosystem.

Costs:

- Less immersive experience.

Plan for Next Two Weeks

1. Reconsider target platform.
2. Reconsider voice chat feature.
3. Design and document high-level systems architecture:
    a. Graphics and UI client sub-system
    b. Voice chat sub-system
    c. Meeting logic core sub-system
4. Update use cases list and diagrams:
    a. Further develop use cases and incorporate rectifications from John.
5. Document technology research, analysis and decisions.
6. Dev Ops research and configuration.
7. Research IBM Watson Personality Insight API (IBM WPI).
8. Research testing strategy.

Links

[1] https://developer.oculus.com/documentation/platform/latest/concepts/dg-cc-voip/

[2] https://www.oculus.com/experiences/go/1101959559889232/