AR Enabled Ground Station

Design Document

Sdmay18-33

Client: Radek Kornicki
Advisor: Diane Rover

Team Members:
Ryan - Communication Lead
Nick P - Deliverables Manager
Jarrett - Assets Aquisition and Scribe
Ethan - Technical Documentation Lead
Nick B - Gitlab Master
Ridwan - Quality Control Lead

Email: sdmay18-33@iastate.edu
Website: https://sdmay18-33.sd.ece.iastate.edu/

Revised: 12/8/2017 V3.0

Table of Contents

1. Introduction

1.1 Acknowledgement

We would like to acknowledge our client, Radek Kornicki, who has provided us with immeasurable support in the forms of technical expertise, funding, equipment, and professional advice has been a key component in our project. We would also like to acknowledge our faculty advisor Dr. Diane Rover, who has helped work as a more functional team.

1.2 Problem and Project Statement

Drones recent skyrocketing popularity has led to an massive increase in business and consumer applications. Many of these applications either require or would benefit from better control of the drone. However, in the current market there is a lack of consumer, and professional equipment that provides the pilot with a better operating experience, especially utilizing Augmented Reality (AR) technology.

Our project seeks to integrate the heads up AR display Google Glass with a custom drone flight controller produced by UAVX. Specifically, our AR display will be able to show alerts, critical data, and general drone status in a efficient and timely manner. This will allow for better control of the drones through increased awareness and ease of use. The pilot will find it significantly easier to find and use the relevant data for their applications, resulting in better piloting and decreased risk of damage to drone or other.

1.3 operational Enviroment

We expect our product to generally be used outside in clear conditions. However, we require that our product must work in any conditions that a drone pilot would be able to fly their drone in. These conditions are generally not very harsh, as drones generally do not fly in inclement weather due to low power output. Our product is not to be used in heavy rain, extreme winds, thunderstorms, tornadoes, tsunamis, volcanic eruptions, and a variety of other natural disasters.

1.4 Intended Users and uses

We intend our product to be used by drone pilots with a wide range of skills, experience, and applications. This has led us to stick to a general platform in regards to UI design, and not focus on too specific applications. However, we do expect our primary consumers to be business operating drones for a business need.

1.5 Assumptions and Limitations

Assumptions:

- The product will be designed for rules and regulation pertaining to the United States
- The product will be within short range of the controller determined (~15 meters)
- The Google Glass will only interface with the UAVX ground controller

Limitations:

- Will not connect to more than one controller at any single time
- Will be kept within normal operating temperatures (-20 C to 40 C)
- The app will be available to all Google Glass users free of charge

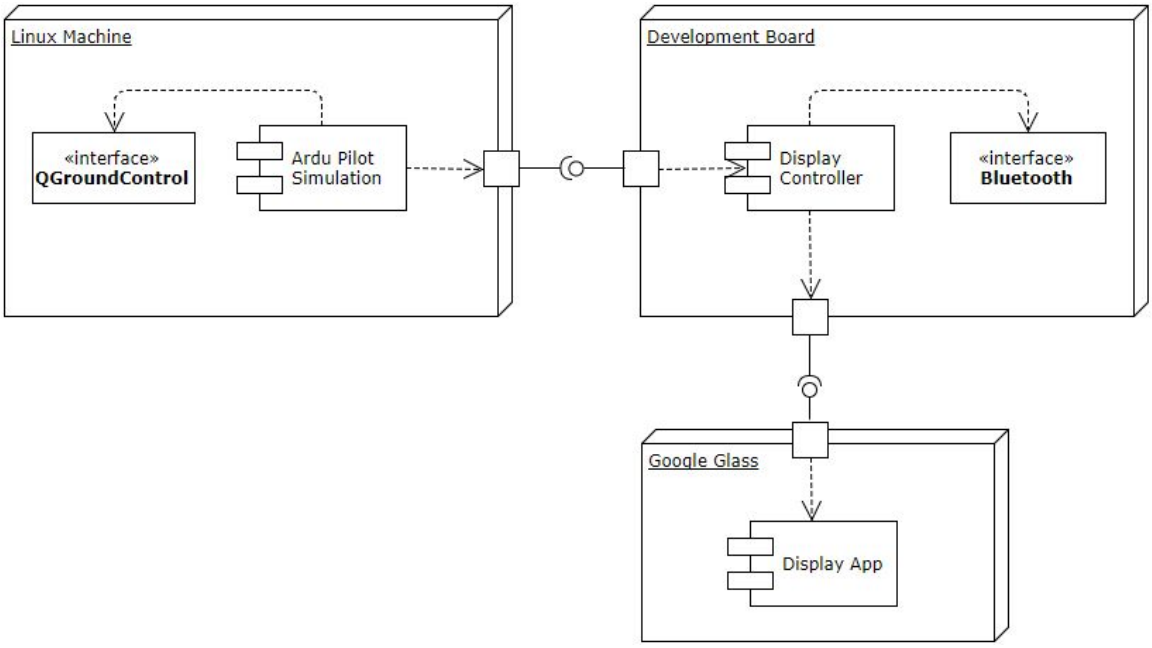
1.6 Expected End Product and Deliverables

The deliverables expected at the end of our project are:

1. A fully functioning Google Glass app that features a clean and responsive UI that provides the user with relevant and timely information from the drone
    a. Expected Delivery: May 2018
2. An Linux app that is used to send the commands from the ground control software over to the Google Glass for processing
    a. Expected Delivery: May 2018
3. The ground station will communicate with the drone over a wireless protocol
    a. Expected Delivery: May 2018

2. Specifications and Analysis

2.1 Proposed Design



Our design has three main components; the linux machine, development board, and google glass. Encased in the linux machine is the simulation system. Ardu Pilot allows us to build a test environment where we can control the conditions, and rigorously test the application with simulation data. This also takes away the physical component, we do not require a drone to complete tests. The linux machine uses an interface called QGroundControl that uses a protocol called MAVLink. The MAVLink protocol is able output various parameters that we can use to do various mathematical computations.

The Second major component is the development board. This board takes in flight data from the Ardu Pilot Simulation, feeds it to the Display Controller, and does computations. The Display controller uses its set business logic to determine when it should push its function calls through the bluetooth interface. One part of the Display Controller logic we have to watch for is the frequency that we push the function calls because a high frequency can be a detriment to the pilot and distract his view of the drone while flying.

Google Glass is the last component. It contains the visual aspect of the project. It takes in the function calls from the development board and pushes it to the display application. Depending on the function call from the Display Controller, the Google Glass will output the specified relevant data to the user.

2.2 Design Analysis

So far we have split up our tasks into two main silos, creating the testing environment and starting to develop a functioning Google Glass App.

On the Google Glass side, we have hit a bottleneck of only having one set of hardware to test with that slows down development significantly. However, in spite of that slowdown we have been able to create a sample application which we all can look at to give us a feel of how Glass development works. Glass development has been fairly challenging for the team but it should go better when everyone is comfortable working in and with Android. Moving forward it would be useful to almost make tutorials for everyone on the team and post them on GitLab so people that are struggling with Glass development have a place to reference when they are having issues.

With the testing environment, we are currently in the process of setting up the environment on a server the ETG provided us. Locally, we have gotten the simulation that we are going to use for testing to run. We have just ran some pre-built simulations and are looking into how they work. On the server we have downloaded and installed Ardupilot, and will be putting QGroundControl on it soon. Besides just running pre-built simulations, we will be reading the MAVLink documentation eventually designing our own simulations and controlling the inputs we will be sending to the simulated drone.

3 Testing and Implementation

3.1 Interface Specifications

There are two main points of interface involved with testing the project, namely the Linux Machine-Development Board point, and the Development Board-Google Glass point. Isolated tests are performed on the Linux Machine before it is ready to interface with the Development Board. Likewise, the Google Glass application will be independently built and tested before interfaced with the Development Board. In order to best understand the scope of testing and implementation within the project, each component will be discussed independently according to current group progress and future work trajectory.

Linux Machine

The Linux Machine was generously provided by ETG's infrastructure in order for SD33 team members to conveniently access the virtual environment through either RDP or SSH connections. This virtual machine runs RedHat and serves as the preliminary environment for bulk simulation and testing. QGroundControl and ArduPilot are installed for the loop simulations of the drone.

BD-SL-i.MX6 Development Board Kit

The BD-SL-i.MX6 board is a low-cost development board built by Boundary Devices. It is equipped with the i.MX6 6Quad application processor that enables development for Linux OS (or Android) multimedia applications. This single-board computer is the functional equivalent of the Ground Station controller developed by UAVX. Displayed in Figs 3.1.1(a & b) is a top and bottom view of the development board. More hardware-specific information on the development board can be found in the reference material listed below.

Figure 3.1.1a

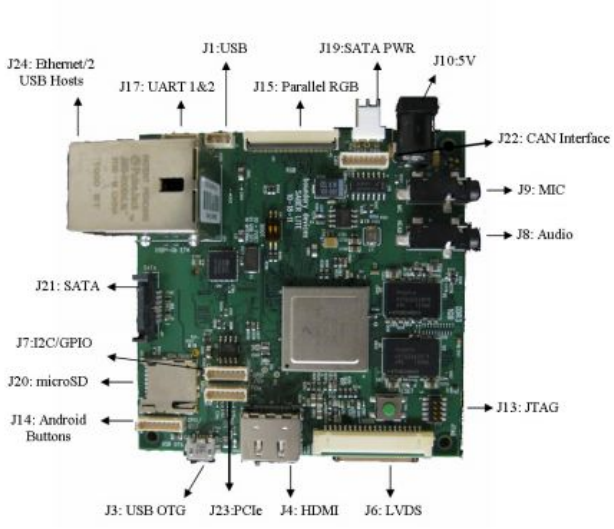


Figure 3.1.1b

BD.SL-i.MX6 Reference Material

Hardware Manual:
https://boundarydevices.com/wp-content/uploads/2014/11/SABRE_Lite_Hardware_Manual_rev11.pdf

Board Schematics:

https://boundarydevices.com/wp-content/uploads/2014/11/sabre_lite-revD.pdf

Google Glass

A main feature of this project is to connect the UAVX developed ground station to Google glass. Google glass released its first version in 2013 as the explorer edition. This is currently the version that we are using and testing with. A new version has been released very recently, named the enterprise edition, and will be used for testing as it becomes available. Information regarding current hardware/software support and editions can be found in reference link below.

Google glass Landing Page:

https://x.company/glass/

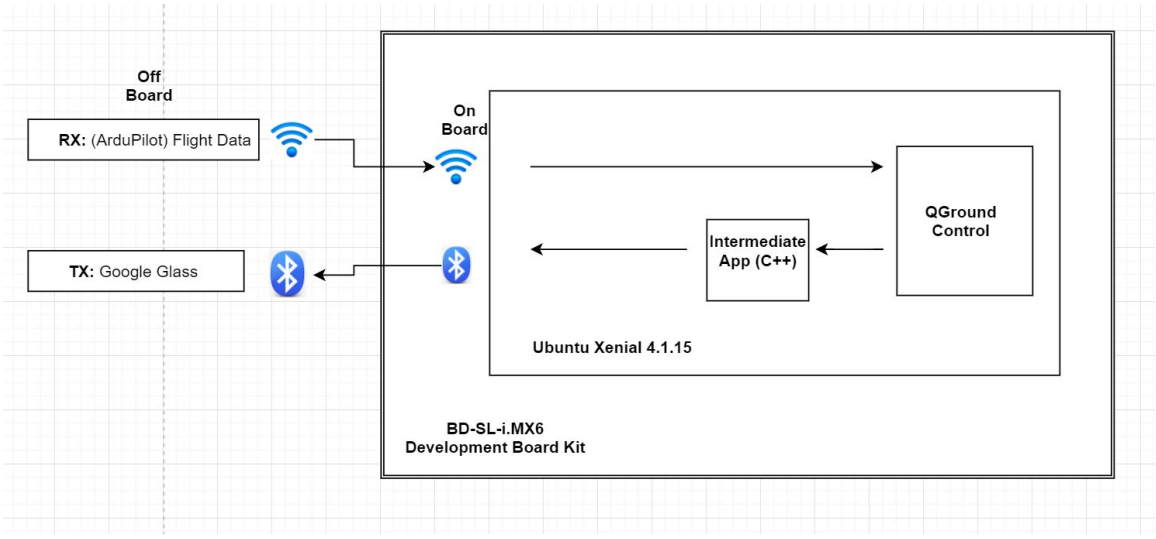
3.2 Hardware and software

In order to streamline testing, we introduced a development board to gain greater control of the NXP/Freescale i.MX6 processor during the development process. This allows us to effectively test our code independent of any hardware concerns that may arise while running on the UAVX ground station while still having access to common peripherals.

QGroundControl and ArduPilot are the two pieces of software are installed for the loop simulations of the drone on the Linux Machine. QGroundControl provides full flight control and mission planning for any MAVLink enabled drone. ArduPilot is the application that will behave as the drone by relaying flight data to QGroundControl. Once the loop simulations are configured and competence is established, the testing will expand to the development board (See Section 3.3).

3.3 Process

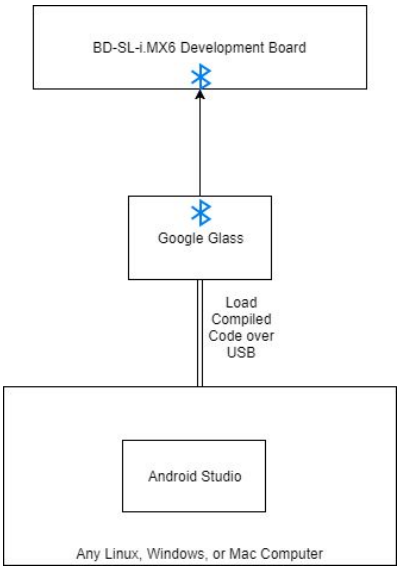
Figure 3.3.1 Development Board Test



Since testing with a drone is dangerous, expensive, and time consuming, we will be using a drone simulation software called ArduPilot. This software outputs the same MAVLink commands that would be outputted from a compatible drone. This software run on a separate laptop or mobile device and sends MAVLink commands over WiFi to the development board. QGroundControl then interprets most of these commands and provides relevant information. Since we can directly access this information on the development board, we can use much of the information that will be routed through an app that we develop to decide on relevant information and send this information over bluetooth directly to the Google Glass.

In order to avoid possible complications imposed over WiFi or bluetooth, we will be using a serial connected at first to ensure that we can correctly process data and slowly implement this over WiFi or Bluetooth when needed.

Figure 3.3.2 Google Glass Test Flow



3.4 Results

Current progress on the development board testing involves running a compatible operating system (Ubuntu) along with the QGC software. This testing will resume once the loop simulations for the Ground Station are configured on the Linux Machine.

We have attempted to configure QGC onto the development board, but have been unsuccessful due to storage issues. In the meantime, we can run all aspects of the simulations on one linux machine since linux is the operating system that will be used on the development board.

4 Closing Material

4.1 Conclusion

We have been able to divide the project into different parts by designing a block diagram that illustrates the overview and flow of the connection of different parts of the project. Also, we have designated each parts to different members based on their strength/skills and interest. For example, we understood the technical needs of the project and divided the team into a front-end/ back end team. A group of people would be working with UI(user interfaces) like the google glass and the backend will be people working on the software which will be installed on the ground station board. In summary, we are setting up our environment and simulating a drone's movements and trying to see if we can collect data from our simulations.

Our overall goal is to make our environment work with a live drone and also establish a communication between the ground-station and the google glass. Our plan is to first simulate this communications on a software application. For example, we have a linux machine that is

controlling the drone simulator by running different commands. After being tested and assured we can establish a communication between the linux running computer and the drone simulator, we will later on install this linux software on the ground station.

We believe this is a great plan because it helps us break the project into its simplest form and working our way up. This, way we can understand the possibilities of any issue that might arise in the future and the kind of information we could gather in a simulation environment before installing them on hardwares(boards). Also, with the different rules surrounding the piloting of drones we thought it better to simulate the functionalities of a drone on a simulator before integrating it with a live drone.

4.2 Appendicies

[1] Google Glass Development

https://developers.google.com/glass/

[2] Ardpilot Software

http://ardupilot.org/plane/index.html

[3] QGroundControl Software

https://donlakeflyer.gitbooks.io/qgroundcontrol-user-guide/content/en/

[4] Development Board Manual

https://boundarydevices.com/wp-content/uploads/2014/11/SABRE_Lite_Hardware_Manual_rev11.pdf

[5] Development Board Schematics

https://boundarydevices.com/wp-content/uploads/2014/11/sabre_lite-revD.pdf