Augmented reality for high-throughput phenotyping

by

Shanshan Wu

B.S., Kansas State University, 2016

A THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Computer Science
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2019

Approved by:

Major Professor
Mitchell Neilsen

Copyright

Abstract

Smart glasses, like smart phones, have separate operating systems, and can execute many different kinds of software and games. Smart glasses can be used to add a schedule, map navigation, interact with friends, take photos and videos, and make video calls with friends through voice control. They can support wireless network access through a mobile communication network.

Bluetooth is a radio technology that supports short-range communication between of the devices. It can exchange information between devices including mobile phones, wireless headsets, laptops, etc. Bluetooth technology can effectively simplify the communication between mobile devices.

This thesis focuses on smart glasses applications for high-throughput phenotyping which requires a data monitor, data synchronization, Bluetooth service, and voice control between devices. On the Android side, the application, which is extended, is called Field Book. The new software called Field Book AR, includes a data monitor module and a Bluetooth server module to achieve data exchange with smart glasses. On the smart glasses side, the application is called DataReceiver. It receives voice commands from users and controls the actions of Field Book AR. Also, when Field Book detects data changing, it accepts new data and shows changes to the users.

Table of Contents

List of Figures

Chapter 1

Introduction

In the information age, agriculture can be very cool and smart. Although agricultural technology is not as sophisticated as aerospace, many people think that data and computing are not so important to the agricultural sector. However, today, in the rapid development of technology, software and algorithms will become more and more important. As smart devices mature, future agricultural technology products will become more lightweight and modular, allowing users to flexibly customize functions according to their needs. A more intelligent knowledge system will provide growers with more convenient and concise technical guidance.

With the improvement of the computing power of intelligent electronic products, the use of augmented reality technology (AR) is more and more extensive. The application in the Internet marketing is more significant, which greatly improves the customer experience. Augmented reality technology is also gradually combined with modern agriculture, injecting new vitality into agriculture. Augmented reality technology is realizing its powerful functions in agricultural display, satellite exploration, and crop growth tracking.

The work described in this thesis is funded by the National Science FoundationâĂŹs Basic Research to Enable Agricultural Development (BREAD) Program along with the Bill and Melinda Gates Foundation. It applies Augmented Reality (AR) technology to the field of agriculture and extends the capability of Field Book[1], an android app used for taking notes on field research plots. Field Book has been developed and released in the Google Play Store and has over 5000 downloads. Both Field Book and this project are open source software. The code can be found on Github. They greatly reduce the cost of high-throughput phenotyping to users and provide reliable convenience.

In this project, we combined smart glasses with smart phone to provide an open source software application group to support a lightweight and targeted function. In fields, users can use Field Book to record and monitor data of crops and plants. When some data changes in Field Book, it will send the change to the smart glasses and allow users to see them immediately. By this function, uses can get the new data very quickly, and make new corresponding observations and judgment on the new data. When a user would like to control Field Book, but they are busy with both hands, they can use voice commands to the smart glasses. Smart glasses can accept the voice commands. After analysis of the command, the smart glasses will send it to Field Book and take the related action on Field Book. This feature frees the user's hands, and provides more convenience and flexibility.

At the same time, we have several projects which is in the same research group. They are Mobile Applications for High-throughput Phenotyping[2], A dynamic, real-time algorithm for seed counting[3], and Extending watershed segmentation algorithms for high-throughput phenotyping on mobile devices[4].

In Chapter 2, enabling technology is discussed. The technology details are introduced in Chapter 3. In Chapter 4, I would like to talk about the challenges of the project and do a simple performance analysis of the software. The future work is introduced in Chapter 5.

Chapter 2

Enabling Technologies

Today, the speed of development can no longer be described as coming. The development of science and technology, economy, scientific research, and other fields is high-speed. When we are admiring a new thing, another new thing is already brewing. Moreover, the speed of renewal is extremely fast, especially the development of electronic technology. Electronic technology has brought benefits and progress to people and society. Its convenience is unanimously recognized by people. Decades of technological advancement, it is making electronic devices thinner and lighter. This provides great convenience for our daily lives. In this project, we use the smart phone and smart glasses as a platform and use Bluetooth as a communication medium, implement targeted practical augmented reality in agricultural.

2.1 Smart Phone

I do not need to introduce smart phones, because everyone knows it. Smart phones can do a lot for us. In recent years, it has also been widely used in agriculture. In the project, we choose the Android phone as a platform. Android is a Linux-based operating system which mainly used for touch-screen mobile devices such as smart phones and tablets. Android is an open source operating system, which could be a low-cost solution. I use Nexus 6 as my test phone. The Nexus 6 was released on October 15, 2014, which was co-developed by Google and Motorola Mobility, and runs the Android operating system.

Figure 2.1: *Nexus 6 smart phone*

2.2 Smart Glasses

Smart wear is a general term for applying wearable technology to intelligently design everyday wear and develop wearable devices, such as watches, bracelets, glasses, and apparel. Wearable smart devices have a history of many years, ideas and prototypes have emerged in the 1960s, and devices with wearable smart devices emerged in the 1970s and 1980s. The advent of the era of wearable smart devices means the intelligent extension of people. Through these devices, people can better perceive the external and their information and can process information more efficiently with the help of computers, networks, and even others. Achieve more seamless communication.

Smart glasses are collectively referred to as wearable glasses devices that have a separate operating system like a smart phone and can be installed by software to achieve various functions. It is one of the most sought-after wearable smart devices that has been proposed in recent years. It is easy to use, small in size, etc. It is widely believed that the emergence of smart glasses will facilitate people's lives. Therefore, it has been researched and developed by Google, Microsoft, and other technology firms. It is regarded as an essential growth point for future smart technology products.

2.2.1 Recon Jet smart glasses

We chose smart glasses to start from Recon Jet glasses which was developed by Intel company. Recon Jet is a sport and fitness glasses, which can share users' metrics to Recon Engage in real time, and then upload them to Cloud and other related functions. It also can be used in industrial production. From the Advertising, it can accelerating employee training, reducing errors and rework rates, and minimizing downtime, fundamentally changing the work flow of industry services such as field service, manufacturing, and logistics. However, when we tried to download the Software Development Kit (SDK) of Recon Jet, it did not support anymore. We gave up developing the app on Recon Jet.

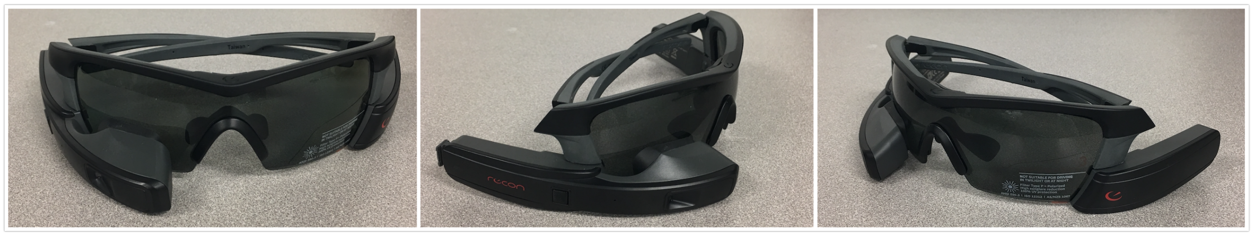


Figure 2.2: *Recon Jet smart glasses*

2.2.2 Vuzix M300 smart glasses

After research on Recon Jet, we started to look for other smart glasses for developing the application. At this time, Vuzix M300 showed up.

The Vuzix M300 is a professional augmented reality smart glasses made by Vuzix, a manufacturer based in the US. The Vuzix M300 offers most of the modern smart phones' features but in a hands-free wearable device. It supports Bluetooth 4.0 connections, which makes it ideal for pairing with Android devices. Also, it can wirelessly connect to the Internet via wifi. Vuzix M300 support voice control, button press and gesture control of touch bar. Users can navigate and use the Vuzix M300 in almost any work environment because of its physical features, like waterproof, dust-proof.

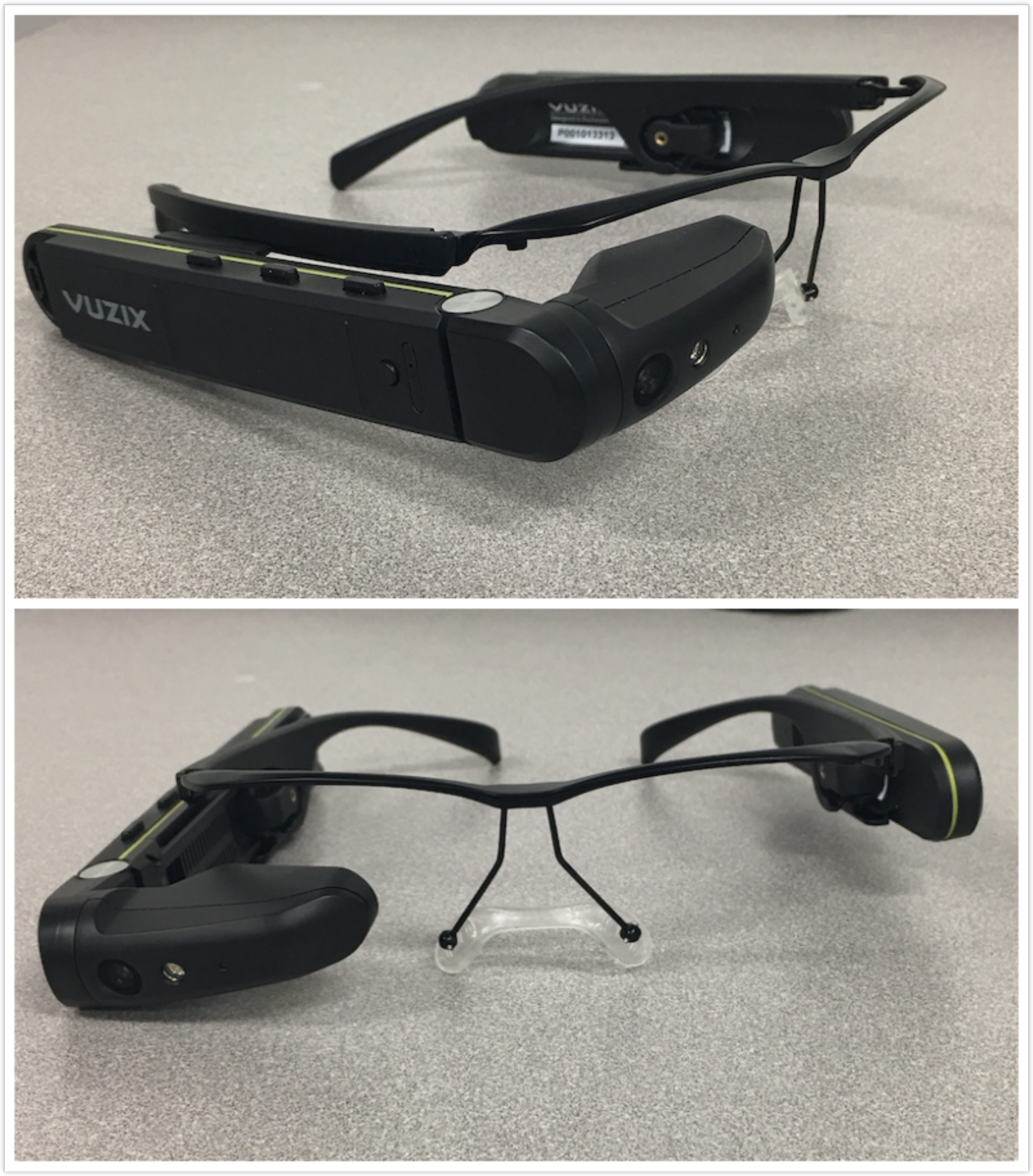


Figure 2.3: *Vuzix M300 smart glasses*

After research on Vuzix M300, we thought we had found the ideal product to develop the project. It integrates all the features which we required. Vuzix M300 supports Bluetooth connection which we use it to implement information communication between smart glasses and smart phones. Vuzix company developed its SDK for supporting voice control which embedded in Vuzix M300. By its SDK, we can define and develop our voice commands in the project. Vuzix M300 runs the Android operating system, which we already familiar with it on Android phones. The development IDE (Integrated Development Environment) is the Android Studio. Finally, we decided to implement the project on the Vuzix M300. Now, the required function of the research already has been deployment on Vuzix M300, which passed the test and all functions work fine.

2.2.3 Vuzix Blade smart glasses

To continue the research, we also got the Vuzix Blade smart glasses in December 2018. Look at the appearance, and it is a standard sunglasses. The Vuzix Blade smart glasses is the CES 2019 innovation award winner. The Blade is designed as a co-worker together with the smart phone so that people can leave their phones in their pocket. The Blade also loads the Android operating system which make it easy to develop apps on it by Android developers. The company reported that more than 400 apps are in the store, which can be download and use.[5] Compare with M300, and the Blade takes a different design. It has the real lens in the glasses. The lens can be prescription glasses, UV protected, etc. There is a small rectangle on the right lens, which is the user interface. The Blade uses waveguide optics and optical principle to project an image on the small rectangle. As same as M300, the Blade also has touch bar and voice commands for control. The Blade plus a remote-control app for both IOS and Android. The highlight of the new feature is that the Blade loads the assistance of Amazon's Alexa.[6]

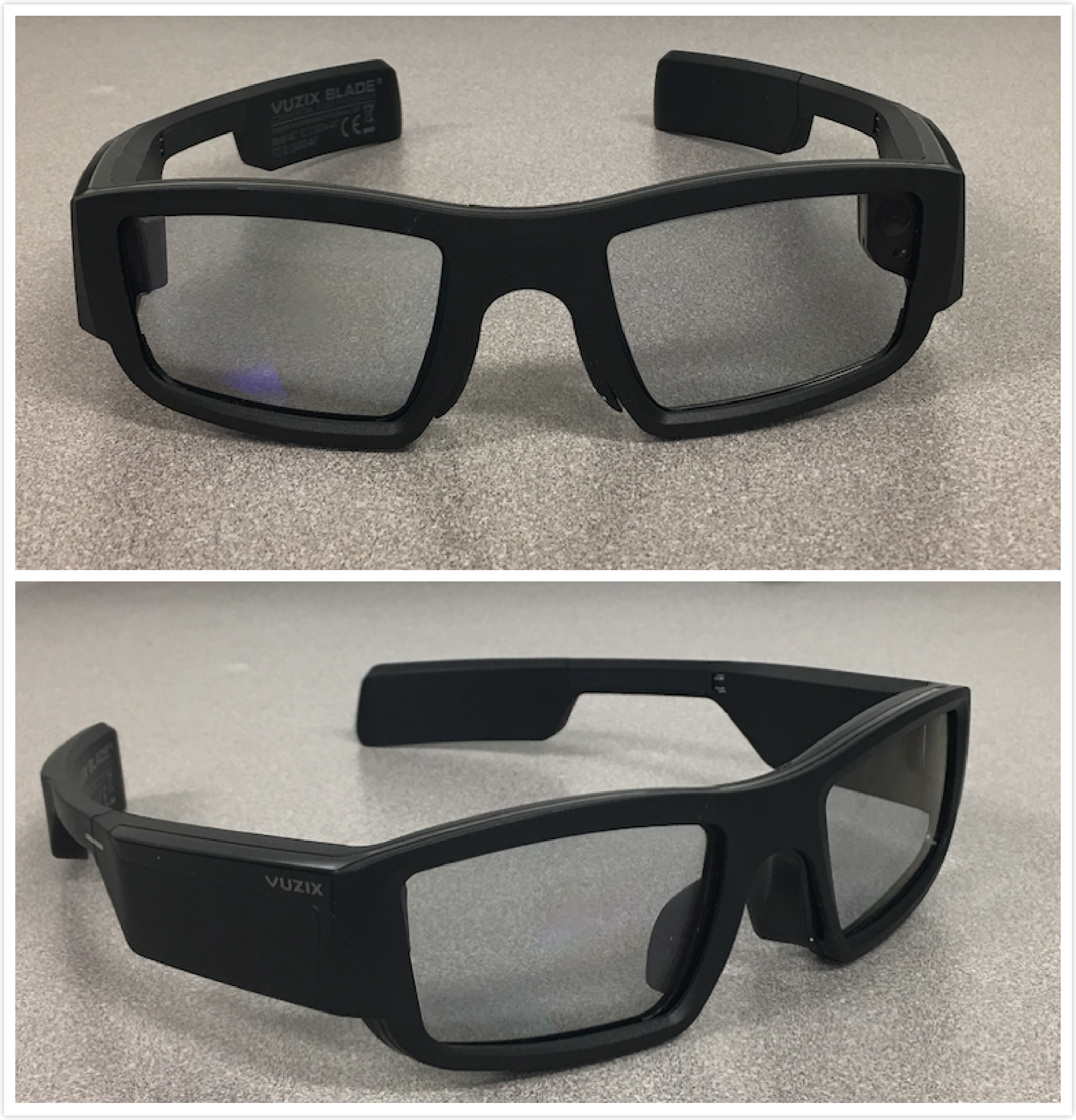


Figure 2.4: *Vuzix Blade smart glasses*

2.3 Bluetooth

Bluetooth is a standard wireless communication protocol based on low-cost transceiver chips with low transmission distance and low power consumption. Since the devices use radio (broadcast) communication systems, they are not connected by actually visible lines, but quasi-optical wireless paths must be feasible.

Bluetooth exists in many products, such as phones, tablets, media players, robot systems, mobile devices, laptops, gamepads, and some high-quality headphones, modems, watches, etc. Bluetooth technology is useful for transferring information between two or more devices nearby under low bandwidth conditions. Bluetooth is often used for mobile device voice transmissions such as Bluetooth headsets or byte data transmission like file transfer for laptops.

On the Android platform, it supports classic Bluetooth and Bluetooth Low Energy. In this project, we choose classic Bluetooth as the data exchange interface between smart glasses and smart phones. The Android platform already provides all required Application Program Interface (API) for Bluetooth data exchange, include Bluetooth setting up, devices finding which are paired devices or available devices of nearby, devices connection, and data transferring between devices.[7]

The Bluetooth Low Energy (BLE) is different from classic Bluetooth, which is part of the Bluetooth 4.0 core specification. Its goal is to provide the lowest power wireless standard and optimized for low power, low bandwidth, low cost, and low complexity. The classic Bluetooth establishes a connection by the socket. The BLE creates a connection based on the Generic Attribute Profile (GATT). After creating a connection, the classic Bluetooth can get and send data between devices by socket API. This is very similar to network communication. In BLE, for implementing data transmission from the master device to the slave device, you need to directly read the acquired slave device's Characteristic and Service which is both UUID form that defined in the slave device. Because they are both called Bluetooth, they have some common points. The obvious one is work flow: setting up devices, finding devices, connecting devices, and transferring data. We have another project OneKK that use BLE. OneKK is an Android app that developed to recode and analyze seeds information, like weight, count, average length and width, etc. The Android platform also provides all the required API for developers.

Chapter 3

Augmented Reality for Field Book

This chapter introduces the details of the project which has two apps, Field Book and Data Receiver. Field Book uses on a smart phone as well as Data Receiver developed on smart glasses. After talking about Field Book and Data Receiver, I introduce the interaction and data flow between two apps.

3.1 Field Book

I have mentioned Filed Book in chapter 1. Field Book is an open source Android app which uses to take phenotypic notes. The data capture of one or multiple traits on each plot is easy and fast because of the friendly user interface design. Compared with paper field book, the app can display additional imported data which users can catch more information at one glance. For each field record, users can choose any additional records of interest to display, like entry names, seed sources, etc. It is an ideal replacement for taking notes on paper. Field Book is open source, and developers can customize new features to meet specific data collection requirements.[1] For providing more convenience, Field Book extends augmented reality technology. When users wear smart glasses, they can monitor data changing in real-time, and also can control Field Book by voice commands.

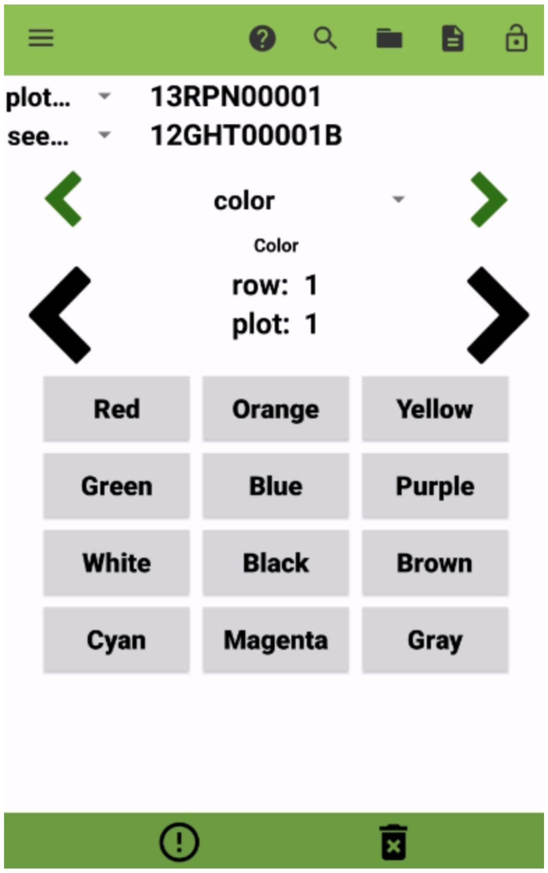
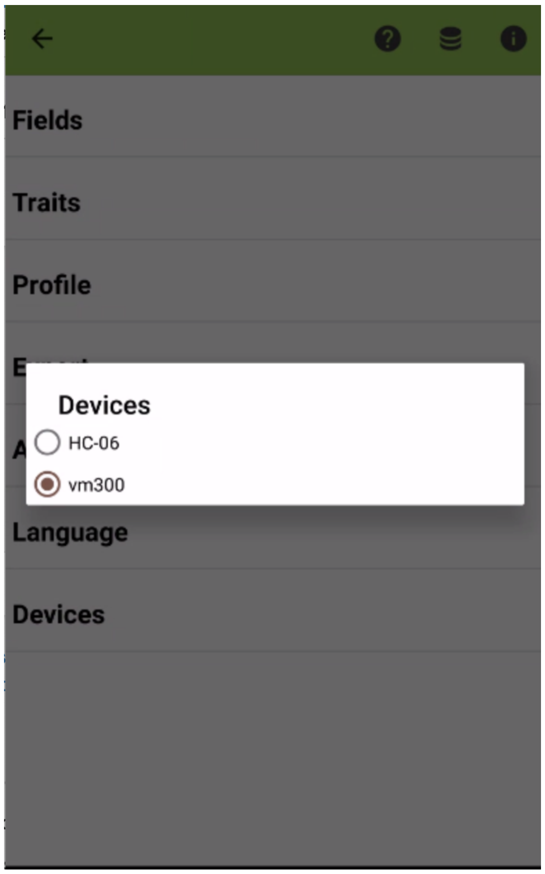


Figure 3.1: *Field Book*

3.2 New Modules in Field Book AR

For developing new features, creating new modules are necessary. I created three new modules for achieving data monitor, and data distributed to internal modules and data exchange with smart glasses. Also adding an option in Field Book setting page to choose the paired Bluetooth device, like Vuzix M300 smart glasses. If users use the same smart glasses, they only need to set it once. If users would like to change a new smart glasses to work, they can reset this setting at any time. The paired device name is stored in the preference file of Field Book. For providing more convenience, Field Book extends augmented reality technology. When users wear smart glasses, they can monitor data changing in real-time, and also can control Field Book by voice commands.



Devices

( ) HC-06

(●) vm300

Figure 3.2: *Setting paired device in Field Book*

3.2.1 Data monitor module

For implementing data synchronization, I create a module to monitor data changing. This module implements several functions to set changing data. In the main class, when data changes because of some reasons like users input, user interface focus on different pages, it calls the functions of the data monitor module. After setting changing data, the data monitor module broadcasts a message to distribute data to the internal module. The message mentions that we have data changing event here. It also attaches the changing data in the message package.

```
public void setRangePlotValue(String r, String p) {
    rangeValue = r;
    plotValue = p;
    intent.putExtra("data", "& plot " + rangeName.charAt(0) + ":
       " + rangeValue + " / " + plotName.charAt(0) + ": " +
       plotValue + " ");

    context.sendBroadcast(intent);
    Log.i("info", "sendBroadcast: " + rangeName.charAt(0) + ": "
       + rangeValue + " / " + plotName.charAt(0) + ": " +
       plotValue);
}

public void setTraitValue(String value) {
    traitValue = value;
    intent.putExtra("data", "& trait: " + traitName + "  value:
       " + traitValue + " ");
    context.sendBroadcast(intent);
    Log.i("info", "sendBroadcast: " + traitName + " " +
       traitValue);
}
```

3.2.2 Data transfer service

Data transfer service module is a service. A service is a component of an application that can run in the background. It does not support the user interface. Usually, a service handles some works to avoid blocking the main thread, because it works in the background. For example, it can deal with network transactions.

Data transfer service is started by the main class of Field Book. Its responsibilities are to filter messages, connect Bluetooth Server to complete data transfer, receive messages to do related actions. When Data transfer service is created, the default action is to register the filtering messages and start Bluetooth Server. After filtering, it only receives data changing message, Bluetooth status changing message and device changing message.

```
@Override
public void onCreate() {

    mActivityReceiver = new ActivityReceiver();
    mIntentFilter = new IntentFilter();
    mIntentFilter.addAction("com.fieldbook.tracker.
    DATA_CHANGE");
    mIntentFilter.addAction("com.fieldbook.tracker.
    BluetoothServer.STATUSCHANGE");
    mIntentFilter.addAction("com.fieldbook.tracker.
    BluetoothServer.DEVICECHANGE");
    registerReceiver(mActivityReceiver, mIntentFilter);

    mAdapter = BluetoothAdapter.getDefaultAdapter();
    mBluetoothServer = new BluetoothServer(this);
    startBluetoothConnection();

}
```

Before starting the Bluetooth server module, it needs the primary setting of Bluetooth. First, we would like to get the Bluetooth adapter. Because the Android platform provides all API of classic Bluetooth, it makes this step very easy. It calls Bluetooth API to get the adapter. The second parameter we wanted is a Bluetooth Device which we already paired and bonded with the current device. The device name is stored in the preference file of Field Book. When getting the bonded device name, the device searches for active, nearby

Bluetooth devices to match the existing device name and bonded device name to acquire the information of the bonded device, like the mac address. After getting all required parameters, it can call the Bluetooth Server initialization function to start Bluetooth Server module.

```java
private void startBluetoothConnection() {
    getDefaultDevice();
    if (mDevice != null) {
        mBluetoothServer.init(mDevice, mAdapter);
    }
}

private void getDefaultDevice() {
    SharedPreferences setting = getSharedPreferences("Settings",
       0);
    String name = setting.getString("BluetoothDevice",
       "DEFAULT");
    if (!name.equals("DEFAULT")) {
        Set<BluetoothDevice> paired =
           mAdapter.getBondedDevices();
        for (BluetoothDevice device: paired) {
            if (device.getName().equals(name)) {
                mDevice = device;
                Log.i(TAG, "Try to creat bluetooth connection to
                   " + device.getName());
                break;
            }
        }
    } else {
        Log.e("DataTransferService", "No boundled device, please
           go to setting choose devices");
    }
}
```

As I mentioned, data transfer service can receive data changing message, status changing message and device changing message. When it receives the data changing message which is from the data monitor module, it acquires the changing data and calls Bluetooth server function to implement data transferring. The Bluetooth connection breaks when it receives the status changing message. This message comes from the Bluetooth server module. At this time, the data transfer service module tries to restart the Bluetooth server. When users change a new device, the device changing message is received. It is requesting to create a new connection. This message is from setting page of Field Book. After getting it, data transfer service cancels and closes the old Bluetooth connection, and create a new connection for the new device. To receive messages, it creates a private class in Data Transfer Service and extends from BroadcastReceiver class.

```
private class ActivityReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        Log.i("Receiver", "Broadcast received: " + action);

        String data;
        if (action.equals("com.fieldbook.tracker.DATA_CHANGE")) {
            data = intent.getExtras().getString("data");
            Log.i(TAG, data);
            mBluetoothServer.write(data.getBytes());
        } else if (action.equals("com.fieldbook.tracker.
        BluetoothServer.STATUSCHANGE")) {

            data = intent.getExtras().getString("message");
            mBluetoothServer.cancelConnection();
            startBluetoothConnection();

        } else if (action.equals("com.fieldbook.tracker.
        BluetoothServer.DEVICECHANGE")) {

            mBluetoothServer.cancelConnection();
            startBluetoothConnection();
        }
    }
}
```

3.2.3 Bluetooth server module

Bluetooth server uses to create Bluetooth connection, accept and send data stream and close Bluetooth connection. After setting the Bluetooth adapter and device information, it starts a thread called AcceptThread to wait for a client device to connect. The purpose of the new thread is to avoid blocking the main thread. During the waiting period, the main thread still works fine.

```
public int init(BluetoothDevice device, BluetoothAdapter
   adapter) {

    mDevice = device;
    mBluetoothAdapter = adapter;

    if (mDevice == null) {
        Log.e(TAG, "Cannot find pairedDevice");
        return -1;
    }
    mAcceptThread = new AcceptThread(mDevice);
    mAcceptThread.start();

    return 0;
}
```

After the client device sends the connecting request, the connection is created successfully, and the AcceptThread is stopped. At the same time, the Bluetooth server creates another new thread called ConnectedThread to receive and send data to the client device. The purpose of ConnectedThread is as same as AcceptThread to avoid blocking the main thread. When the input data stream is voice commands, the thread broadcasts it with the action information. The main class receives the broadcast. When the ConnectedThread input data stream run into the exception, it broadcasts status change message which means socket disconnect. Data transfer service receives this message and tries to re-create Bluetooth connection.

```
public void run() {

    mmBuffer = new byte[1024];
    // bytes returned from read()
    int numBytes;

    // Keep listening to the InputStream until an exception
       occurs.
    while (true) {

        try {
            // Read from the InputStream.
            numBytes = mmInStream.read(mmBuffer);
            String s = new String(mmBuffer);
            Log.i(TAG, "Receive message from DataReceiver: " +
               s);
            mIntentVoiceCmd.putExtra("voiceCmd", s);
            mContext.sendBroadcast(mIntentVoiceCmd);

        } catch (IOException e) {

            Log.d(TAG, "Input stream was disconnected", e);
            mIntentSender.putExtra("message", "Socket
               disconnected");
            mContext.sendBroadcast(mIntentSender);
                                      break;
        }
    }
}
```

3.3 App in Smart Glasses

We selected Vuzix M300 smart glasses for this project. It loads the Android operating system, so the development process is the same as development of the Android phone app. The app in Vuzix M300 is called Data Receiver because the original function in smart glasses is only received the data stream and show them on the user interface. With the perfection of this app, it also sends data to the server device and accepts voice commands now.

3.3.1 Bluetooth client

Bluetooth client uses to create a connection with the server device. After setting Bluetooth adapter and device information, it creates a thread called ConnectThread to connect server device. At this time, if the server is ready to be connected, the connection can be created at once. It sends a message to the main class to claim the connection created. Users can get it by the user interface. If failed, it sends a failed message to the main class. Main class sets it on the user interface to hint the user. Users can reset the server device information or check if the server device is opened and try to re-connect. When a connection is created, ConnectThread is stopped at once. Bluetooth Client starts a new thread call ConnectedThread to accept and send the data stream to the server device. The purpose of both threads is to avoid blocking the main thread and make sure the main thread works fine during the connecting process. Compare with Bluetooth server, and the client uses Handler to pass messages. The reasons are that Bluetooth client only needs to pass messages to the main class, as well as Bluetooth server need to pass messages to multiple modules. Sending messages by broadcast is more suitable for Bluetooth server.

3.3.2 Voice control module

Speech recognition technology, also known as Automatic Speech Recognition (ASR), aims to convert vocabulary content in human speech into computer readable input such as binary codes or character sequences. Recent years, speech recognition technology is a popular conception. It provides great convenience for our lives. Several well-known IT companies have launched their speech recognition products, like Amazon's Alexa, Apple's Siri, etc. The Android platform also provides related API to support speech recognition development on Android devices. In this project, we use Vuzix voice recognition SDK to achieve voice control development in Vuzix M300.

Voice control is an essential function in the project. With this feature, users can easily control devices. In the voice control module, we mainly support 4 voice commands for now. They are "Trait", "Plot", "Next" and "Back". Trait and plot are mode names, and they correspond to the trait and plot in the Field Book. After setting mode, users can say "Next" or "Back" to execute next or back of trait or plot dialog in the Field Book. So the user would first say "Trait" or "Plot" which would persist as a saved variable. Then they could say "Next" or "Back" to move forward or backward in the list of traits or entries. For traits, this would be equivalent to pressing the green arrows of Field Book, and for plots, this would be equivalent to pressing the black arrows of Field Book. When the voice control module is created, it calls the function of Vuzix speech client to enable voice recognizer and register voice commands. Anytime users say the voice commands which already been registered can be recognized. After handling the voice command, it is sent out as byte format by Bluetooth client module.

```
public VoiceCmdReceiver(MainActivity activity) {

    mMainActivity = activity;
    mMainActivity.registerReceiver(this, new
        IntentFilter(VuzixSpeechClient.ACTION_VOICE_COMMAND));

    try {
        VuzixSpeechClient sc = new VuzixSpeechClient(activity);
        sc.deletePhrase("*");
        sc.insertPhrase("Trait", TRAIT);
        sc.insertPhrase("Plot", PLOT);
        sc.insertPhrase("Next", NEXT);
        sc.insertPhrase("Back", BACK);
        VuzixSpeechClient.EnableRecognizer(mMainActivity, true);

    } catch(NoClassDefFoundError e) {
        e.printStackTrace();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

3.4 Data Flow

The primary function of this project is to monitor the data changing of Field Book and show them to customers in the smart glasses. There are several ways to trigger data changing event, like user input to Field Book, user input to Data Receiver (voice commands). According to the trend of the data stream, the data flow is divided into active and passive. Active data flow is triggered by user input to Field Book. The data flow started with Field Book user interface and ended with Data Receiver user interface. The passive data flow is triggered by voice commands. It starts with Data Receiver, passes to Field Book and is back to Data Receiver at last. The active data flow is Simpler and more intuitive than passive data flow.

3.4.1 Active

The active data flow means the data changing caused by Field Book on the smart phone. It could be the user input or dialog switch in the user interface. When data changes, the Bluetooth server sends data out. Bluetooth client receives it and shows it in the user interface of smart glasses. The data flow shows on the following diagram.

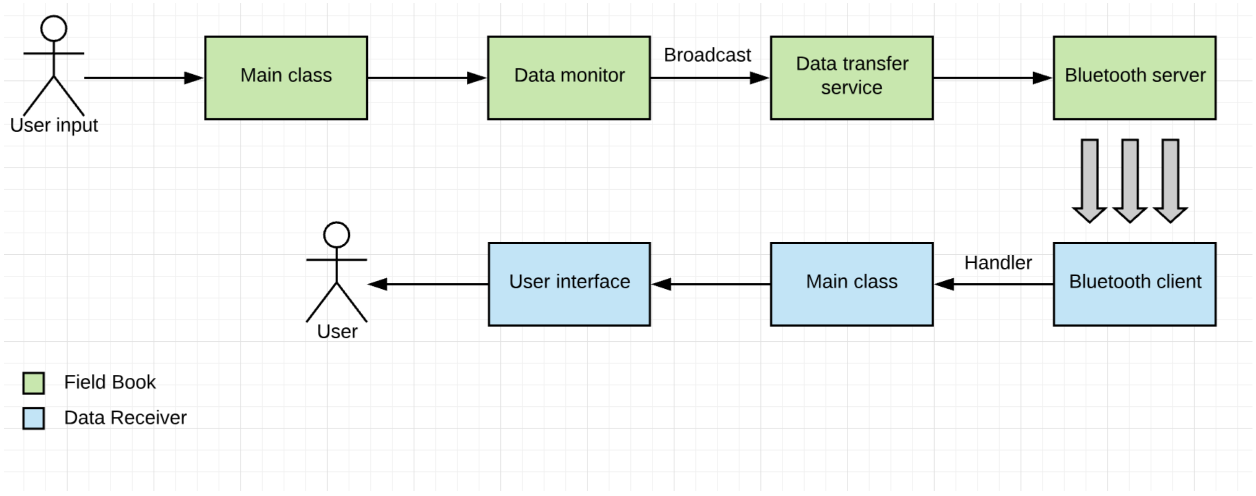


Figure 3.3: *Active data flow*

3.4.2 Passive

The passive data flow means the data changing caused by the Data Receiver of smart glasses. Users release the voice command, and voice command receiver module accepts it. After analysis, it generates the text command and calls Bluetooth client to send out. On the smart phone side, the Bluetooth server accepts the message. Moreover, send it to the main class by broadcast at once. The main class changes the user interface depends on the command messages. Because the user interfaces changing, the data monitor module is called at this time. It triggers the data changing event. After data transfer service receives the event, it calls Bluetooth server to send the changing data out. On the smart glasses side, Bluetooth client receives the changing data. It passes the data to the main class by Handler, and the main class shows the data on the user interface. The data flow shows on the following diagram.

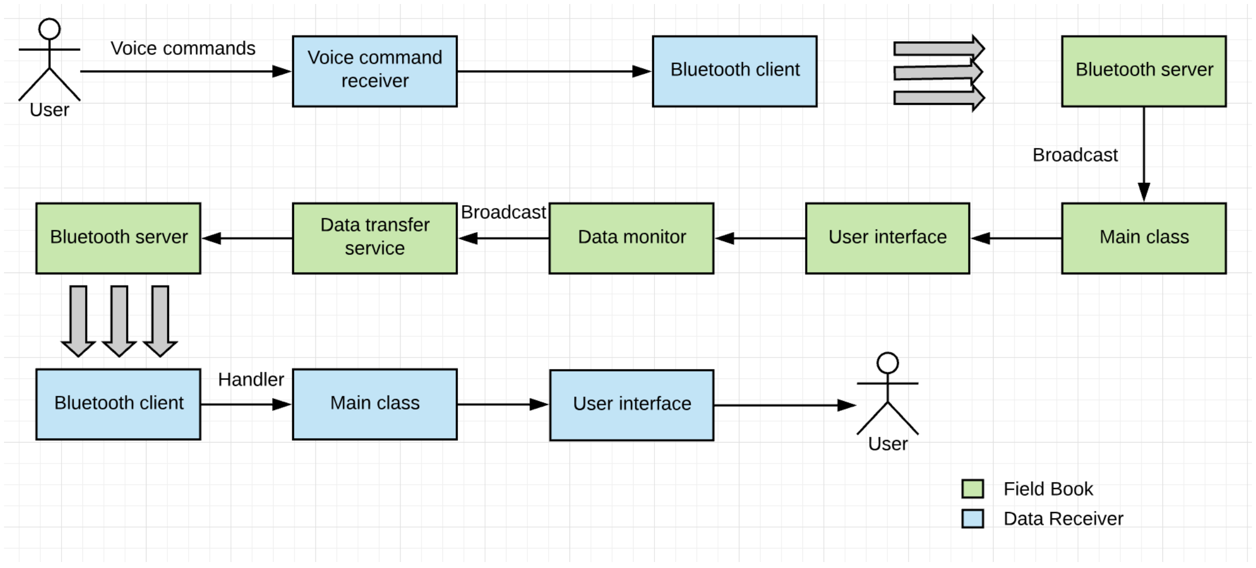


Figure 3.4: *Passive data flow*

3.5 Smart Glasses App Customer Guide

The user interface on smart glasses is different from the user interface on smart phones. We control smart phones by touching the screen. However, we cannot touch the screen of smart glasses. The smart glasses designed to use touch bar combined with one or two buttons to be controlled. Because of it, I designed the user interface of the Data Receiver as simple as possible. Users can easy to control it by touch bar.

As same as Field Book, when users use Data Receiver for the first time, they need to set the bonded device. In Data Receiver, it has a toolbar at the top of the main page which can show the current connection status, voice command mode and set bonded device. It looks like the following picture.

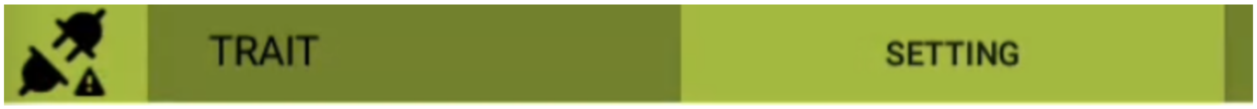
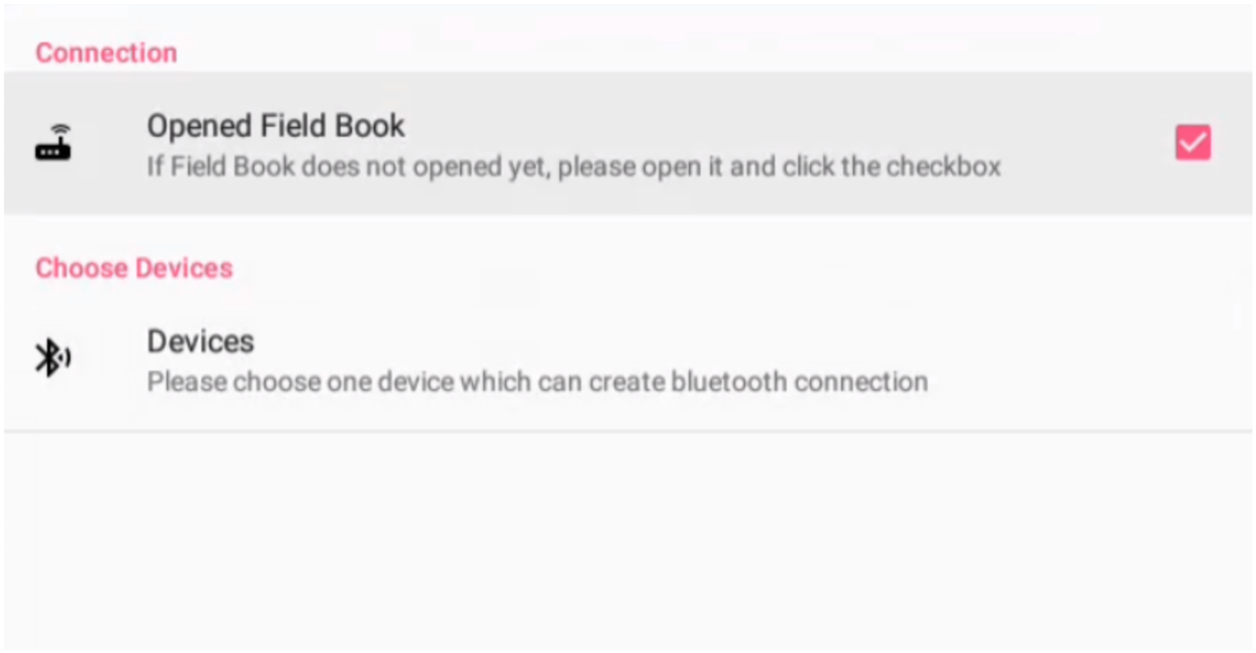


Figure 3.5: *Disconnect toolbar*

The leftmost sign means it disconnects between smart glasses and smart phones now. The "TRAIT" part is the voice command mode. There are two modes, trait and plot. It is in trait mode now. Trait is the default voice command mode. The "SETTING" part is a button. User can click it to show the setting page. The setting page looks like the following picture.


Connection

Opened Field Book ✓

If Field Book does not opened yet, please open it and click the checkbox

Choose Devices

Devices

Please choose one device which can create bluetooth connection

Figure 3.6: *Setting*

The disconnected status could be caused by setting the wrong bonded device or Field Book is not open which means Bluetooth server is not ready. If it still disconnects after users set the correct bonded device, we would consider the Field Book is closed now. Users should open the Field Book and check on "Opened Field Book" option to hint the Data Receiver that it should be all right now. The Bluetooth client tries to re-connect the Bluetooth server when the setting page is reset. The device choosing page is similar to the Field Book device choosing page. If users do not change the connected device, it only needs to do the setting for the first time to use the Data Receiver. The device name is stored in the preference file of the Data Receiver. Every time users start the Data Receiver, it reads the preference file and acquires the bonded device name. After that, it tries to connect the bonded device, and displays the connection status on the toolbar. The device setting page looks like the following picture.

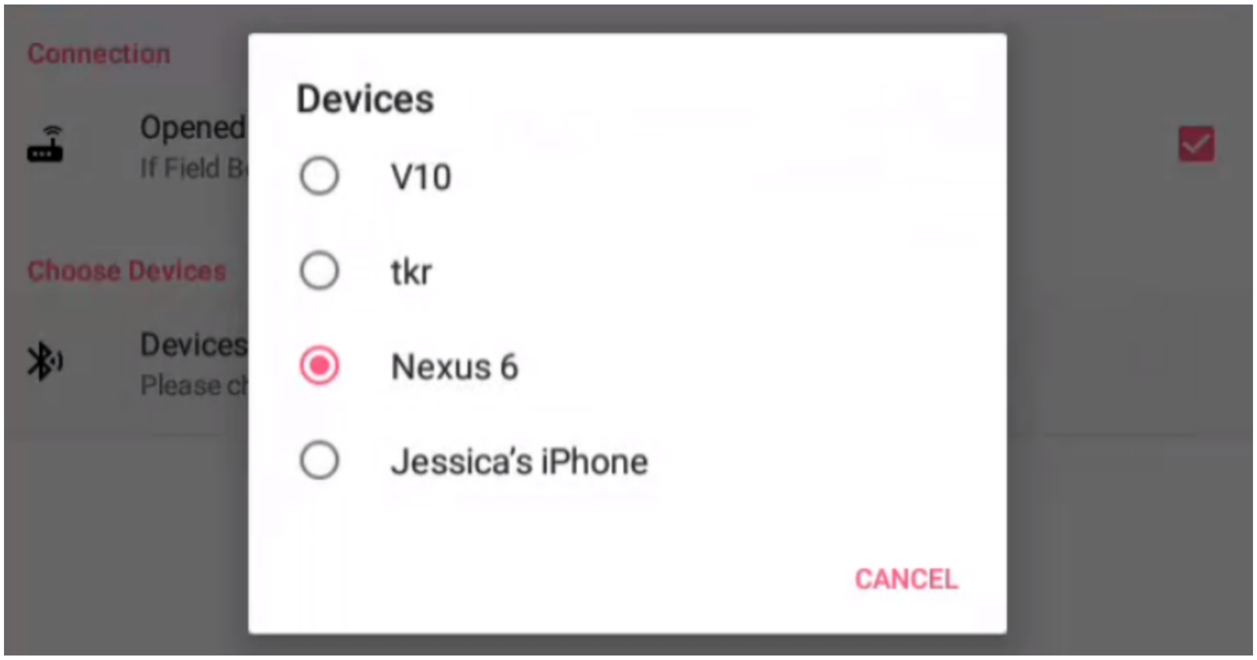


Figure 3.7: *Device setting*

When getting the setting page down, users can go back to the main page. The toolbar shows the connection create successful by the leftmost sign. If users would like to change the connected device, they can get the setting page to set at any time.

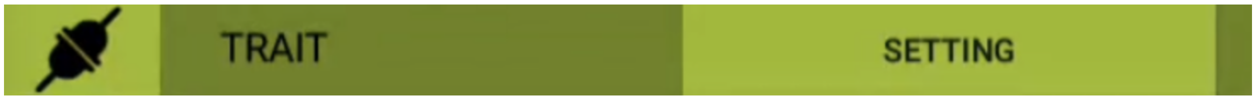
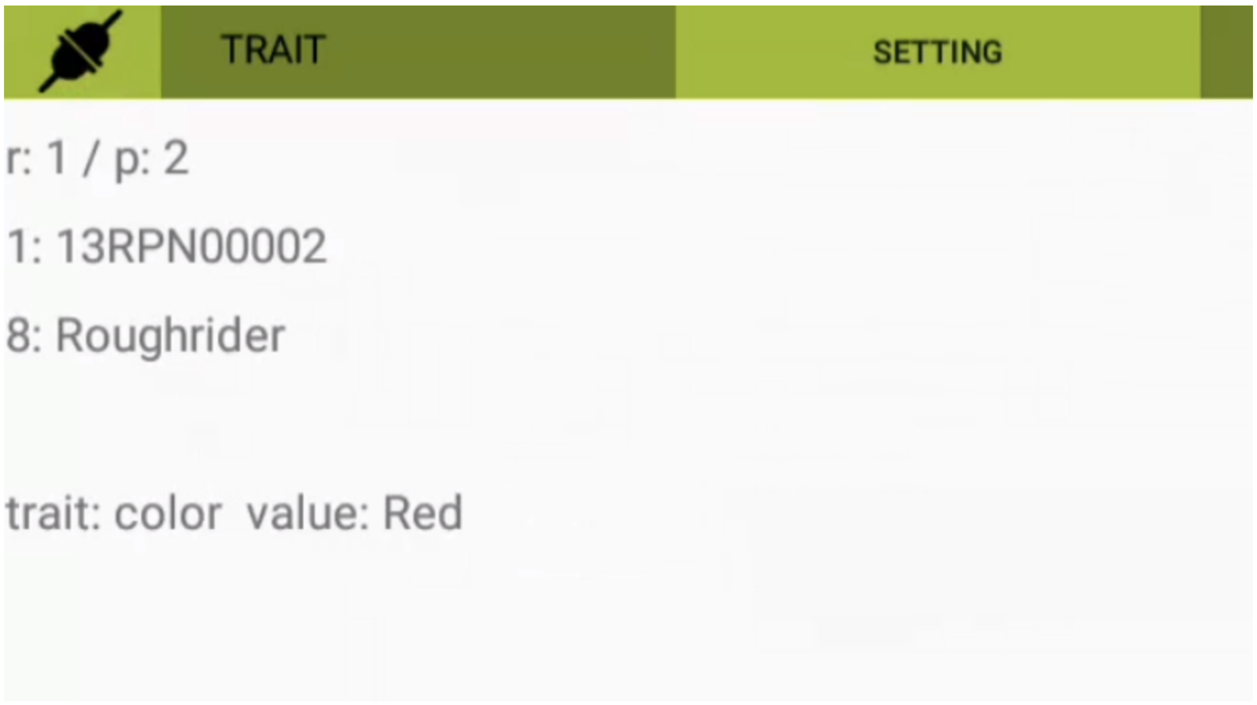


Figure 3.8: *Connected toolbar*

After the connection created success, data transferring can be started at any time. When users input on the Field Book or user release voice commands to Data Receiver, the main page of the Data Receiver shows the syncing data to users. For now, there are four lines of data. The first line is the current number of row and plot. The second and third lines are the value of plot and range. The number displayed on the second and third lines correspond to the information bar of Field Book. The last line is trait and its value. The main page looks like the following picture.



r: 1 / p: 2

1: 13RPN00002

8: Roughrider

trait: color value: Red

Figure 3.9: *Data Receiver main page*

Chapter 4

Analysis

4.1 Technical Challenges and Solutions

When I started this project, I was a beginner in Android development. The Bluetooth related development was my first time to try, either. I met several challenges when I started this project. My way is to cut the big target into small steps, complete them one by one. My first step was to learn the essentials of Android development, like how to deploy a user interface, how to connect a user interface and Java code, etc. To achieve this goal, I watched some Android development videos, followed their steps to finish several small Android apps. I also read the official Android document and studied the examples on the official website which provides excellent help during the development process.

4.1.1 Create the Bluetooth connection

The first challenge is to create the Bluetooth Connection between smart phones and smart glasses. When I finished the Bluetooth server and client code, I tested them. The suppose result should be data exchange success, but it failed. I did so much research online for failure reasons and still cannot get a good solution. Finally, I re-read the official document and found that the Universally Unique Identifier (UUID) between Bluetooth server and client should be set to the same UUID in classic Bluetooth. After setting the same UUID, the problem resolved.

4.1.2 Receive multiple messages at the same time

When the data changing event happens, the changing data is sent out by Bluetooth server which happens on the smart phones side. On smart glasses side, Bluetooth client receives it and show it to the user interface. If there are so many data changing events that are triggered at the same time, the changing data is sent out by Bluetooth server one by one. However, the messages could be received at one time on Bluetooth client side, which can cause data analysis errors or data missing. For solving this problem, I add an interval symbol at the start of each message. When the Bluetooth client accepts messages, in knows how many messages it has received by the interval symbol count, and it analyzes them one by one. In this way, it avoids missing data or analysis errors.

4.1.3 The Vuzix voice control SDK

When I started the voice control part, Android speech is my first choice, because of Vuzix M300 loads Android operating system. I downloaded several speech demos to study. When I installed speech example to M300, it did not work. I read M300 instruction and found that it has its SDK to support voice control. I installed Vuzix SDK to Android studio and started to learn how to use it. The process is the same as studying Android speech, read documents and study example code. Finally, I got it to work. In Android phone, we usually say "ok, google" to trigger the voice control. In Vuzix M300, we need to say "Hello, Vuzix" to trigger voice commands.

4.2 Performance Analysis

Android studio provides the Android Profiler to measure app performance. Users can learn how the app uses CPU, network, memory and battery resources by the Android Profiler. It provides real-time data to users and shows them by a dynamic diagram. The following shortcuts show the usage of CPU and memory of Field Book and Data Receiver.

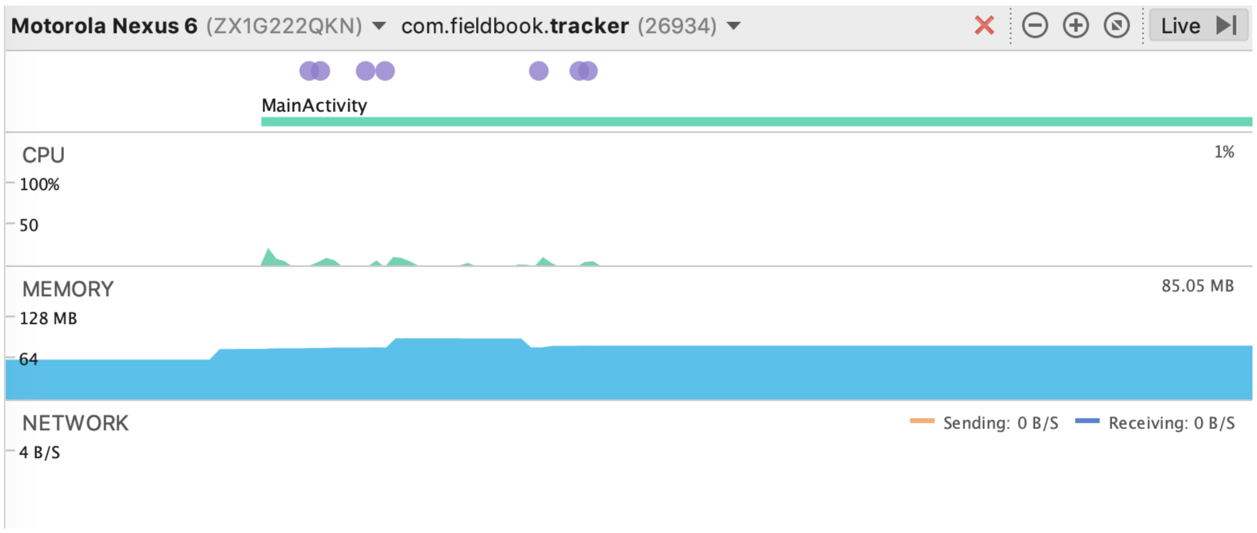


Figure 4.1: *Field Book usage of CPU and memory*

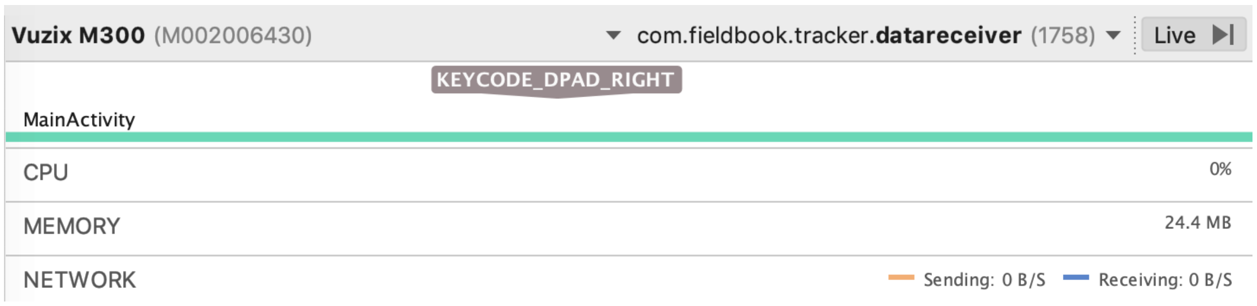


Figure 4.2: *Data Receiver usage of CPU and memory*

From the diagrams, the CPU usage of Field Book and Data Receiver are excellent. The memory usage of Field Book is higher than Data Receiver. The reasons could be it takes up more resources than Data Receiver. For example, the layout of Field Book is more complicated than the Data Receiver. Field Book also has a background service module. Moreover, it supplies the storage of collecting data by SQLite which is the database of the

Android platform. Data Receiver has a simple layout, and it does not have the background program. It doesn't store data to the database.

For the data exchange part, there is no delay when the smart glasses accept the changing data from Field Book. For voice control, it takes 1 to 2 seconds to analysis the voice commands by smart glasses after users release it, then it is sent to Field Book. Users can observe the result from the user interface of Field Book after they release voice commands 1 to 2 seconds. For creating a Bluetooth connection, it connects very quickly if the configuration is appropriate.

Chapter 5

Conclusions

5.1 Summary

This paper described the implementation of Date Receiver which is a research project at Kansas State University, which introduced the background technologies and the application of the project. Data Receiver is an extension of Field Book which deploys on smart glasses. It provides the basic AR module in agriculture application field. The project supports the data transferring between smart glasses and smart phones. It provides data synchronized in real-time. Voice control also implemented in this project. People can control Field Book indirectly by voice control because smart glasses accepted the voice commands and sent to the smart phone by Bluetooth connection.

5.2 Future Work

We want to continue the research and extension of the current AR app. There are several points can be implemented in future work.

5.2.1 Looking for cheap smart glasses

As I mentioned before, the apps in this research are all open source. People can install them for free. Also, developers can download the source code and customize their features. Android is an open source platform. People can get an Android phone at a low cost. In the project, only the smart glasses are the expensive device. If there are low-cost smart glasses, it would be perfect.

5.2.2 Support more voice commands

Now, smart glasses only accept specific voice commands. Add more voice commands could be a more important future work. We have talked in chapter 2, and voice commands can free users' hands. Users can focus on more critical observation or investigation of things. We can add voice notes function, and users can use voice commands to record notes instead of input by smart phones keyboard.

5.2.3 Capture images by smart glasses

Capture images by smart glasses also can be future work. Now, Field Book is to take photos for fields or plants and store them to the database. Users can capture images by smart glasses and send them to Field Book. When the user wears smart glasses, he/she only need to twist the head to let the camera focus and release the voice command to take a picture. The process is cool and convenience. It fully meets the purpose of smart glasses, liberating users' hands and providing great convenience to users.

Bibliography

[1] T.W. Rife and J.A. Poland. Field book: An open-source application for field data collection on android, 2014. URL http://www.wheatgenetics.org/field-book.

[2] Mitchell Neilsen Chaney Courtney and Trevor Rife. Proc. 30th intl. conf. on computer applications in industry and engineering (san diego, ca, oct. 2-4, 2017), mobile applications for high-throughput phenotyping, 2017.

[3] Siddharth Amaravadi Zhiqiang Xiong Jesse Poland Mitchell L. Neilsen, Chaney Courtney and Trevor Rife. Proc. the 26th intl. conf. on software engineering and data engineering (san diego, ca, oct. 2-4, 2017), a dynamic, real-time algorithm for seed counting, 2017.

[4] Shravan D. Gangadhara Mitchell Neilsen and Siddharth Amaravadi. Proc. the 30th intl. conf. on computer applications in industry and engineering (san diego, ca, oct. 2-4, 2017), extending watershed segmentation algorithms for high-throughput phenotyping on mobile devices, 2017.

[5] Mike Prospero. Vuzix blade review: These 1,000 dollars ar glasses are fun but frustrating, 2019. URL https://www.tomsguide.com/us/vuzix-blade,review-6065.html.

[6] Corey Gaskin. Vuzix blade and m300 hands-on, 2018. URL https://www.phonearena.com/news/Vuzix-Blade-and-M300-hands-on_id101589.

[7] Bluetooth overview. URL https://developer.android.com/guide/topics/connectivity/bluetooth.

Appendix A

Source Code on Smart Phone

Field Book AR code can be found on Github: https://github.com/Jesssica33/Field-Book

Appendix B

Source Code on Smart Glasses

Smart glasses app code can be found on Github: https://github.com/Jessssica33/DataReceiver