Home Automation: Development of Application with Google Glass

Lalita Peersingh Purohit¹, Dr. Kalpana Sharma²

¹Computer Application, Bhagwant University
²Computer Science and Engineering, Bhagwant University

Abstract—In current era researchers developed such wearable devices that have opened new capabilities in human interactions with the world. For the application of wearable devices, home automation is an area of interest. Now home automation is done with mobile devices and home control panels. This paper presents an implementation of home automation using the Google Glass. The user can monitor the home command and control home appliances with a few simple voice commands or finger swipes, thus bringing a new level of interaction with the home. The Glass communicates with an online server that controls commands to a microcontroller to perform certain tasks. The microcontroller, in turn able to communicate sensor information to the Glass through the same server. This framework serves as a prototype that executes the potential for wearable technology to simplify the users in home experience. In approaching future it can be expanded to include more features and more devices for the user to interact with in the home.

Keywords—server; commands; automation; control; environment; etc.

I. INTRODUCTION

Google Glass has a touchpad which allows a user to swipe through different panels called ‘cards’. These cards can show you the weather, your recent emails and even show you directions around a city. All of this is fed back to the user through a small display in the upper right corner of the user’s field of vision. In the medical field Google Glass has been adopted widely because of its technology. The hands-free aspect, combined with its camera and WiFi capabilities, make it a powerful tool to train and educate people. For example, a surgeon could theoretically record an operation and a medical student could watch that recording all the whilst taking notes in preparation for medical examinations. The fast advancing of modern devices has led to the growth of new ways for people to interact with the world. One of these advancements is wearable technology, which joins the power a smartphone with convenience and accessibility. Wearable technology offers computing devices that can be worn, like a backpack or a belt, and have form of input, such as a touchpad [1]. The rise in demand of wearable technology opens up a new wealth of possibilities for potential applications that includes sensing the user’s immediate environment [1], navigating [1], and assisting medical professionals [3]. The application of wearable devices is in home automation. The primary purpose of home automation is to optimize the efficiency and comfort of home residents. However, present efforts to fulfill these desires face limitations ranging from restricted portability, such as that provided by wall panels, to a shortage of a nearly hands free experience when using tablets and smartphones [3]. Wearable devices mitigate these problems with constant accessibility and awareness of their context, or the user’s surrounding area [1]. As it is constantly accessible, the interfaces of most wearable devices are designed to favor regular use [5]. Google glass which focuses on efficiency and simplicity can fulfill these necessities. It allows users to simply complete their daily tasks through advanced voice and motion recognition. Moreover, the Google Glass is convenient because it can be worn like ordinary glasses, providing a novel experience. This research designs an innovative approach to home automation through Glass application development. The key
features of our application, “Start home auto”, Arduino, incorporate the Android Software Development Kit (SDK), and the Google App Engine. In the medical field Google Glass has been adopted widely because of its technology. The hands-free aspect, combined with its camera and WiFi capabilities, make it a powerful tool to train and educate people. For example, a surgeon could theoretically record an operation and a medical student could watch that recording all the whilst taking notes in preparation for medical examinations. This is useful because Glass does not have a keyboard for manual entry. To use the MyGlass app users need a Google account. Once signed into MyGlass, setting up Glass is a few taps. MyGlass also is available online where a user can enable or disable various services like Google Now, Google Books, YouTube, etc. One of many things that the MyGlass app can do is enable Glass to connect to a WiFi network.

II. BACKGROUNDS

2.1 Why Google Glass?
It is a wearable device with an optical head mounted display and a touchpad on the right temple of the glasses. Interaction can be done by Google Glass both by using voice commands and by tapping/swiping the touchpad. In the top-right of the user’s field of view Glass displays information, in a location that is easily accessible without obstructing the user’s sight. Besides this Glass can connect to the internet through Bluetooth and Wi-Fi, and can use this to access servers and devices.

2.2. Components
The development of this application relies on several programs, hardware and tools. The Android Development Toolkit (ADT) allows development of android mobile applications on the Google Glass. The Google App Engine, a “Platform as a Service” (PaaS), allow the android application to access an online server. Eventually, the Arduino microcontroller, an open source electronics platform, controls most of the hardware.

2.2.1. Android Development Toolkit
Google Glass works on the latest Android’s operating system, 4.4.2, which is also known as “Kit Kat.” This Android Development Kit (SDK) includes libraries, a debugger, a handset emulator, sample code, tutorials, and documentation. To write Android applications the Java programming is used. The extensible Markup Language (XML) is also used for design in the ADT. XML is used to declare static structures like that in the Android Manifest. The Manifest holds the application name, version and icon (not applicable with this application) also permissions from the operating system, used to request additional features of a device’s hardware or access personal information and services. The official supported integrated Development Environment (IDE) IS Eclipse. The Glass Development Kit (GDK), a revision to the ADT, is used to code specifically for the Google Glass. The GDK handles specific Google Glass functionality, such as voice commands. In addition the entire project is compiled with GDK and targets the Google Glass instead of the Android phone operating system.

2.2.2. Google App Engine
This PaaS allows developers to build and run applications through Google. A PaaS is a category of cloud computing that allows for the building of applications and of services upon the internet. The utilizations of the app engine are for applications that have web service components, applications that sync across devices and keep a profile, and applications that need authentication. App Engine Endpoints API parses parameters a JSON (JavaScript Object Notation), a lightweight data interchange format to translate data between servers. Google App Engine was used to create servers. The servers allow Arduino to communicate with the application via the internet and vice-versa. The Google app engine is specially formulated for this application because it is specifically designed for interfacing between android or Google products or servers. The App engine API allows
comfortability for controlling the input and output to and from the server. Its annotation-based note provide easiness of utility from a developer’s perspective.

2.2.3. Arduino
It is a single board micro controller that senses the environment by receiving senses from many sensors. It can also make output data and can be programmed, allowing it to control devices. While there are various types of Arduino models the choice of controller is Arduino Yún because it is optimized for wireless connection, which is crucial to the application. The Yún also contains a micro slot card, USB-A port, 20 digital input/output pins (of which 7 can be used as PWM outputs and 12 as analog inputs), a 16 MHz crystal oscillator, a micro USB connection, an ICSP header, and 3 reset buttons. Arduino Yún is programmed with a C/C++ based language using the Arduino IDE.

III. DEVELOPING A GOOGLE GLASS APPLICATION
Start Home Auto was formed to show the potential of wearable devices in home automation. The application allows the user to control their home appliances. This project contains three components: hardware controlled by Arduino micro-controller, the Google glass application, an online server hosted on App engine.

3.1 Glass Application
The Google Glass runs on the Android operating system, and is also programmed using the Android Development Toolkit (ADT) in the Eclipse development environment. It uses the voice command “Start Home Auto,” the user can open the application and view the home screen, a menu that allows the user to access the toggle menu, live stream-video, or programed menu. These options can also be used to wirelessly control monitor the home hardware through Live-streaming and motion-detection.

3.2 App Engine Servers
Google App Engine Server lets the Arduino and Google glass application to interact. The communication flows in both directions. The Glass application can send a command, as a “lights on,” to the server. The command is then relayed on the Arduino, which performs the actual command. When the Arduino detects motion, it will prompt the server, and the server thereby provides a timestamp for when the motion was detected. The Glass application polls the server periodically and when it finds that motion is detected, it will get the timestamp and notify the user.

3.3 Arduino Microcontroller and Hardware
An important feature of the Arduino is that you can create a control program on the host PC, download it to the Arduino and it will run automatically. Remove the USB cable connection to the PC, and the program will still run from the top each time you push the reset button. Remove the battery and put the Arduino board in a closet for six months. When you reconnect the battery, the last program you stored will run. This means that you connect the board to the host PC to develop and debug your program, but once that is done, you no longer need the PC to run the program. The Arduino microcontroller is an easy to use yet powerful single board computer that has gained considerable traction in the hobby and professional market. The Arduino is open-source, which means hardware is reasonably priced and development software is free. This guide is for students in ME 2011, or students anywhere who are confronting the Arduino for the first time. For advanced Arduino users, prowl the web; there are lots of resources.
A switch is toggled by which is directly connected to the hardware, namely the lights and the air conditioner. With the output received from the Google App Engine, the Arduino turns the devices on or off. Besides this, the Arduino is connected to an infrared sensor, which serves as a motion detector. When there is fluctuation in the values of the infrared sensor, the Arduino indicates the server and then pauses before again checking the sensor.

**IV. RESULTS AND DISCUSSION**

![Figure 1. Project Schematic](image)
4.1 Application Activities and Classes
The finished application has a simple user-interface that involves some activities. The activities included are a Toggle Activity, a Manage Tasks Activity, an ADD Tasks Activity, and a Live Streaming Activity. In the actual code for the application, there are a total of eleven classes. The detected motion has been informed to user by Notification Activity. Otherwise there is no separate screen for this notification. The remaining classes communicate with the Google App Engine Server. Table demonstrates the various uses for each of the classes in the application.

4.2 Application Expandability
The current prototype of Start Home Auto and associated hardware involve abilities to toggle air conditioning and lights to notify the user of motion. The user can also create a task program that will itself execute the toggling of air or the lights conditioning at a preconfigured time. When the user navigates to the Toggle Menus, he or she can tap the Google Glass to bring up the Toggle Menus, which will give the current status of the air and lights conditioning and allow the user to toggle between states by tapping. Whenever the Arduino detects motion, Google Glass will alert the user and give the time and date it was detected while the Google Glass is on. The user can take appropriate action. The current framework of the sharing information from Google Glass to the Arduino and vice versa is a framework that can be prolonged to include a wide variety of capabilities and tasks, which simplifies the user experience. For example, we can program Google Glass to tell the Arduino to open the garage. The Google Glass can then track the user’s location thereby enabling the user to configure the application to open the lights or garage when the user within a certain radius of his or her home. The Arduino, or any microcontroller, can receive data from sensors around the home. With appropriate hardware and utilizing the comfort of Google Glass’s voice commands, the user can ask Google Glass to do even more. For instance, a coffee maker can be instructed to prepare coffee every time the user asked Google Glass, “Glass, make me coffee.”

![Figure 2. Glass User Interface Diagram](image-url)
4.3 Using Start Home Auto
Whenever the user turns on the Google Glass, he or she can view the applications using the voice command “ok glass.” Upon getting the menu, the user can speak, “start home auto” to run the application. This command takes the user to the main activity. The main activity is a list including three options: Program, Toggle and Stream. Using voice commands again or taps on the touchpad permits the user to choose an option. If Toggle is chosen, then the user has two screens, which he alters to send information to the server. The first is the light toggle, with the option to change the status of the “Lights” to “ON” or “OFF” with the tap of the touchpad. The second option on Toggle is the AC toggle, which functions with the same method. The user again has two options if the programme is chosen: Add Tasks or Delete Tasks. Tasks include programming the application to automatically toggle (either the lights or the AC) at certain times. In the future, such an activity can also be programmed to toggle assured tasks when the user is a certain radius away from the home. If Stream is selected at the main activity, then live stream from the security camera feed is displayed. The application checks for motion detection every minute from the server. When motion is detected, Glass notifies the user with the time detected. Figure 2 reflects the Glass User Interface (UI).

4.4 Challenges Faced
Several factors made designing and implementing home automation using Google Glass difficult. Recent updates to Android’s operating system introduced bugs in the latest version of the Eclipse IDE, which meant that it would have to be returned to an older version. The Google Glass, being a fairly recent development, did not have all the tools available for ease of development. For example, the lack of an emulator made testing Glass code difficult. Additionally, the Arduino Yun, designed especially for Wi-Fi connections experienced problems accessing the internet and the secure server. Next challenge was using Glass’s simple interface to program some complexed aspects of the user interface. For example, selecting a time on a phone is slight due to the screen size (which is larger than that of glass) and its touch screen. Glass, on the other side, uses a smaller screen, only a touchpad, and a simplistic design philosophy. As like to choose a time when programming tasks, a user must navigate through four menus, first, a menu containing all the hours, then a menu containing all the “ten minutes” (the tens digit of the minutes), the “ones minute,” and lastly, whether the time is AM or PM.

4.5 Future Employment
The idea of home automation using wearable devices is easily stretchable, and this project demonstrates just a fraction of its capabilities. “Smart Home Auto” was created to be built upon a model for future applications. Its abilities are less as a result of time constraints and limited access to home devices. Home automation systems should maximize user customization. “Start Home Auto” in some ways limits the extent to which the user may personalize the application for his certain needs. The application permits the user to toggle only two devices. Applications that will build upon these features should allow the user to more easily add devices and program more features with comfort. After all home automation application systems will eventually be more accessible to the public. Because the application depends on its users to have some background in code to understand its works and use it perfectly. It is not entirely practical. In future applications will build upon its functions and current capabilities as a first of glass home automation applications. The next generation of applications of its kind will make a home automation application that is more accessible and user-friendly to consumers.

V. CONCLUSIONS
Start Home Auto serves as a platform for upcoming projects involving the integration of the home with wearable devices. The Google Glass proved as best medium for exhibiting the potential of
wearable devices because it incorporated simple convenience with powerful computing unmatched by other kinds of wearable devices to date. The simple interface which includes a menu to toggle functions, program tasks, and stream video, captures the three of the most basic features of the integration of such technology with the home. The user can control his or her home appliances and monitor the surrounding atmosphere without as much as moving a finger. Apart from this the motion detection serves as an example of how appliances in the home can communicate to the wearable device. Flexibility that is inherited in the program allows for an easy expansion of functionality for more practical applications.

REFERENCES


VI. https://conservancy.umn.edu/bitstream/handle/11299/168229/GoogleGlass_Ellingson.pdf;sequence=1

VII. http://www.ge-ip.com/blog/google-glass-manufacturing/


