AUTOMATED ANSWERING APPS FOR PEOPLE WHO ARE VISUALLY-IMPAIRED USING GOOGLE GLASS

Harsh Seth

Submitted to the faculty of the University Graduate School in

partial fulfillment of the requirements for the degree

Master of Sciences

in the School of Informatics and Computing,

Indiana University

May 2015

Accepted by the Graduate Faculty, Indiana University, in partial fulfillment of the requirements for the degree of Master of Sciences.

Masters Thesis Committee

Professor David J. Crandall

Professor Sriraam Natarajan

Professor Gregory J. E. Rawlins

Harsh Seth

AUTOMATED ANSWERING APPS FOR PEOPLE WHO ARE VISUALLY-IMPAIRED USING GOOGLE GLASS

People with visual disabilities have to face numerous challenges in their everyday lives. Current assistive technology uses a "human powered access technology" approach, in which people use their camera-enabled mobile phones to take images of their environment and send them to human workers on the internet, in order to help them identify objects and situations. But people who are visually challenged often find these technologies difficult to use, error prone, slow, and limited in scope. In this thesis, we investigate and build applications on Google Glass using computer vision techniques to automatically answer photo-based questions asked from people who are visually challenged. The first application answers questions related to people in the vicinity of the user who is visually impaired. The second application helps visually challenged users to identify objects based on personalized tags. Testing of both applications illustrates that they are easy-to-use and have better speed and accuracy compared to existing applications like VizWiz [8] which use humans as workers. Our experiments evaluate how reliable Google Glass can be as an assistive device.

Table of Contents

Chapter 1. Introduction 1
1.1 Assistive technology for visually impaired people 1
1.2 Issues with assistive technology 3
1.3 What is wearable computing? 4
1.4 Google Glass and our work 7
1.5 Outline of the thesis 9
Chapter 2. Methodology 10
2.1 Why Mirror API? 10
2.2 Mirror API Architecture 11
2.3 OAuth 2.0 14
2.3.1 The OAuth dance 15
2.3.2 Accessing resources 17
2.4 Workflow of the applications 18
2.5 Application 1: Face detection 20
2.6 Application 2: Object recognition 24
Chapter 3. Results 28
3.1 Face detection results 28
3.1.1 Dataset 28
3.1.2 Experimental protocol 30
3.1.3 Accuracy results 31
3.1.4 Timing results 32
3.2 Object recognition results 38
3.2.1 Dataset 38
3.2.2 Experimental protocol 38
3.2.3 Accuracy results 38
3.2.4 Timing results 42
3.3 Discussion 42
Chapter 4. Conclusion 44

4.1 Summary 44
4.2 Future Work 44
References.....46

List of Figures

Figure 2.1 Mirror API architecture 12
Figure 2.2 Data flow in the Mirror API 13
Figure 2.3 Installing and using Glassware 15
Figure 2.4 Redirect to Google for authorization 16
Figure 2.5 Using access token to issue Mirror API request 17
Figure 2.6 Workflow of Glassware 18
Figure 2.7 Sharing captured image 20
Figure 2.8 Adding a caption 20
Figure 2.9 Read aloud cards 23
Figure 2.10 Paginate response cards 23
Figure 2.11 Example 1: Counting the number of people in front of the user 24
Figure 2.12 Example 2: Finding the age of everyone around the user 24
Figure 2.13 Sample questions supported by the system 24
Figure 2.14 Cards showing object tagged as coke 25
Figure 2.15 The train command card sent to the API 26
Figure 2.16 Response for search instruction 27
Figure 2.17 Response for recognize instruction 27
Figure 3.1 Performance of individual categories 31
Figure 3.2 Evaluation of questions 32
Figure 3.3 Response time for application one 32
Figure 3.4 Sample of dataset from face detection Glassware 37
Figure 3.5 Fraction of questions answered correct vs fraction answered 39
Figure 3.6 Sample of dataset from object recognition Glassware 40

List of Tables

Table 3.1 Results of face detection 33
Table 3.2 Results of object recognition 41

Chapter 1: Introduction

1.1 Assistive Technology for Visually Impaired People

There are 285 million visually impaired people in the world: 39 million are blind and 246 million have low vision [15]. These people confront numerous challenges in everyday lives such as accessing information, navigating to places, interacting with environment and people, etc. [4, 6]. Effort has been made to address these problems with the help of assistive technology [4, 6, 7, 8]. Hersh [16] states that Assistive Technology aims to bridge the gap between what people who are visually challenged want to do and what the existing social infrastructure allows them to do. One can thus conclude that assistive technology helps visually impaired people to have full and equal participation in all aspects of society [16].

Vel azquez [4] mentions that core areas of research on assistive technology for people who are visually challenged have been in information transmission and mobility assistance. The author states that the area of information transmission includes reading instruction or texts, identifying products and their brands, recognizing colors, etc. The development of reading devices which would particularly help in information transmission is still at an early stage. The author also talks about the second area of mobility assistance,

which is more complex and challenging. It involves scanning the surroundings of the user, helping the user in the real world, avoiding obstacle, etc. [4].

Advances in technology have made mobile devices with assistive technology very common amongst the visually impaired people [17]. Ye and et al. [17] in their work report that visually impaired people feel safer in public with a mobile phone. Touchscreen devices with screen-reading software like iPhone and Android have gained popularity with visually impaired people [17]. There is a lot of research being conducted on accessibility of touchscreens, which involve solutions in both hardware and software. Software solutions involve applications on phones that provide interaction with the screen through audio output, and hardware solutions include vibrations on the phone as a haptic feedback to convey braille [18]. Along with screen reading capabilities, mobile phones are also equipped with screen-magnification, hearing aid compatibility, and hands-free operation [19].

Assistive applications on mobile phones have started to replace specific-purpose devices like barcode readers, and compasses [6, 17, 19]. These applications can be categorized into two types. The first type of apps include automated programs using computer vision to solve problems like currency recognition, optical character recognition, navigation, and shopping [20]. The second type of applications is based on human computation or "crowdsourcing" [9] like VizWiz [8], where the visually impaired people click images, ask questions about the image, and receive answers from remote human workers in near real time [6]. The advantage of using "human-powered assistive technology" is that there are no restraints on the type of questions that the visually impaired people can ask about the image [5].

1.2 Issues with Assistive technology

A lot of studies and research have been conducted to understand the reliability and issues of assistive technology [9, 10, 17, 21]. A study by Kane et al. [21] observe that visually impaired people feel assistive technology with specialized hardware devices such as talking barcode readers, and color identifiers, etc. are expensive and have limited capabilities. There are studies which have examined the accessibility problems of mobile devices related to hardware and software [22, 23, 24, 25]. Kane et al. [21] in their work talk about how the form factors of mobile devices, such as small keys, thin size, small on-screen text and no tactile buttons, makes it inaccessible and inefficient to operate the device by people who are visually challenged [17]. For instance, people who are visually challenged find it difficult to operate their phones while walking, mainly because one of their hands is pre-occupied by a walking cane or trained dog, or because the audio of the screen reading app can interfere with their hearing for navigation.

Borodin et al. [26] asked visually challenged people about screen reader browsing techniques and they concluded that there is information that the screen reader app cannot convey to the user, making the application error-prone and unreliable. Ye et al. [17] show that visually impaired people often find it cumbersome to operate their mobile phones due to the screen reader app, primarily because of the noise made by the app. People who are visually impaired feel that screen reader apps do not have the capability to let them operate their phones discreetly under certain situations such as in a meeting, funeral, etc. [17]. Under these circumstances they are forced to either use headphones or lower the volume of their mobile devices in order to maintain social etiquettes [17].

Another area which remains unsolved and has been overlooked while developing assistive technology for the visually impaired people has been privacy [10]. For example, a major privacy concern for visually impaired users is eavesdropping. Many users tend use to screen magnifiers on their devices which makes it easy for people nearby to read [10]. Even screen reading apps are audible enough to bystanders, who can easily eavesdrop [10]. Although the users can use headphones to listen to the device, this completely cuts them from their surroundings, making them vulnerable to other dangers in their environment [10]. Another privacy concern regards the pictures that are being shared with crowdsourcing apps. These images may contain private content that the user does not intend to share like credit cards, bank statements, etc. There is no method on the app to check whether there is private content, in the image making it risky to use the application.

Considering the flaws mentioned above in devices that support assistive technology, one can say that visually impaired people need enhancements to these devices. As per Ye et al. [17], people who are visually limited need devices which are easy to carry, easy to operate, easy to access and are discreet in terms of privacy. They need a device which does not attract attention while performing tasks [17], but new technology in the form of wearable devices could offer one possible solution to these requirements.

1.3 What is Wearable Computing?

The trend of hardware shrinking in size and computation power increasing has stirred a lot of changes to the prevailing concept of a computer [14]. Computers have reduced from the size of a mainframe to the size of mobile phone and wrist watches. By the late 1990s, we could make the size of this hardware small enough that it became feasible

to place "computation power on a user's body making it accessible at all times" [14]. This led to development of a whole new range of devices coined as "Wearable Computers" [14]. Wearable Computers were devices that were always working, always accessible and easily worn on the body.

There have been many attempts made to define Wearable Computers. One of the first notions was made by Thad Starner in 1993 in his work [27]. The author suggested that Wearable Computers should have two major characteristics: persistence and consistence. The author explains that persistence means the wearable interface is constantly available and can be used concurrently with other tasks. For consistency, the author describes that devices hardware and functionality should remain the same in every situation.

As the field developed, there were more definitions added to the term Wearable Computers. Bradely in his work [28] describes Wearable Computers to have five properties. He states that Wearable Computers should be portable while in operation; enable hands-free and hands-limited use; get users attention even when not in use; are always on, sensing users' context to serve better. Starner in his thesis [14] mentions how most of these definitions only concentrate on the ideal interface of the device and avoid talking about the how the hardware should be implemented for the devices. He believes that an ideal wearable computer should persist and provide constant access, indicating that the device needs to be "mobile and unobtrusive" [14] to achieve this goal. The author states that the device must try to observe and model the user's environment, the user's physical and mental state, and the wearables own internal state. This would reduce the effort of the user to access the system. Finally, he states that an ideal wearable computer should interact seamlessly by adapting to existing "input and output modalities automatically to those

which are most appropriate and socially graceful at the time" [14]. From all the definitions stated above, one can conclude that the purpose of wearable computers is to provide continuous, seamless, convenient and portable access to computers.

In 1977, a camera-to-tactile vest was the one of the first wearable devices developed for people who are visually challenged. The device converted images from the surroundings to a 10 inch tactile grid on the vest [29]. WatchIt, a wristwatch computer particularly useful for people who are visually impaired, had pressure sensors embedded onto it allowing the user to scroll through an audio menu [17]. Pasquero et al. [30] developed wristwatches which could be paired with mobile phones to provide notifications via haptic output [17]. Navigation has been the core area of focus while developing wearable computers for the people who are visually challenged [4, 31]. The different navigation systems developed include obstacle detection, providing directions to the user via audio or haptics output, and interpreting 3d world to a 2d haptic surface [17].

A survey conducted by Ye et al. [17] showed that 70% of visually impaired participants agreed to use Google Glass as a wearable device that will help them visualize the environment and communicate about it via audio and vibrations. The participants voted speech (55%) as the most preferred output modality followed by haptics (24%), both of which can been seen in Google Glass. Recently, Ahmed et al. [10] surveyed visually impaired people about ideas for applications on Google Glass and applications such as counting number of people in a room, detecting how close they are to the user, etc. were suggested. These findings inspired us to choose Google Glass as a device for developing applications for visually impaired.

1.4 Google Glass and our work

With current issues in assistive technology, wearable devices like Google Glass provide opportunity for assisting visually impaired people. The Glass is designed in such a way as to not require its use to look down, operate keys, navigate through complex menus, or type frantically [11]. Glass, in an aesthetically elegant form factor, integrates capabilities such as capturing first-person images, detecting touch, bone conduction transducers for audio, voice commands, communication, searching, and processing [32]. This may make Glass easy to carry, operate, access and provide discreet operation in terms of privacy.

Googles stated goal for Glass was to reinvent the Human-Computer interaction, and to design it in such a manner that Glass requires minimum input from the user to negotiate the system [11]. Glass takes input either by voice commands or from the trackpad built in the system. This could make entering text on the device, accessing apps, capturing images, making phone calls, searching the web, etc. very convenient for the people who are visually limited. Although Glass has a prism display as form of output, it is also capable of reciting the text content on the screen to the user, satisfying the speech output requirements for visually impaired people. The audio for output reaches the user through bone conduction transducers. This means the user can listen to text on screen without use of any headphones or speakers, which not only helps them keep their device discreet but also keeps their ears free to focus more on their surroundings.

The Glass is also a “hands-free, ears-free and wires-free” [11] means of staying connected to the internet, which implies the visually impaired can access any information from the internet, anytime through Glass. This idea gave further motivation to our work: to make most of the internet.

Our focus was on developing applications on the Glass using computer vision to answer photo-based questions from the visually impaired user. Through these apps we try to demonstrate how Google Glass can outperform the mobile and other assistive devices currently being used, in terms of performance and privacy. Taking inspiration from the survey conducted by Ahmed et al. [10], we started building a simple app to perform face detection. Since Google Glass does not support face detection internally, we used third party API knowns as Orbeus Rekognition Application Programming Interface (API) [12], to perform the task. The API returns a response containing successfully detected users in the surroundings along with a few additional features such as age, emotion, gender, etc. We then equipped the app with the capability to answer questions from the visually impaired users, related to these features. They could ask questions such as "how many men are present around me?" or "How many happy people are there around me?" and so on. This application can help the users who are visually challenged know more about the different type of people they encounter in a social environment.

The second application aims to help visually impaired people identify objects that they use in their daily life, based on personalized tags. For this app we again make use of the same third party API, Orbeus Rekognition [12]. Users who are visually impaired can photograph images of objects, that they use regularly or have difficulty identifying, and tag them with the help of the app. They can then send a "train" command to the API, which triggers the API to learn about the objects and their respective tags. After this training, whenever the user faces trouble in identifying objects, they can photograph the unknown object and ask the Glassware to identify it with the help of the API. This application can

be useful in many scenarios such as finding the correct can from the kitchen cabinet, identifying brands while shopping, etc.

1.5 Outline of the thesis

The chapters in this thesis discuss in detail about Google Glass and the applications we developed on it.

Chapter 2 provides details about the architecture for the Mirror API and our applications, and application workflows.

Chapter 3 presents the results and tests performed on the applications.

Chapter 4 provides conclusions and directions for future work.

Chapter 2: Methodology

As Firstenberg and Salas [11] mentioned, Glass is a "hands-free, ears-free, wires-free" medium of staying connected to the internet. Since Glass is a wearable device with limited computational power, we need to leverage internet services to perform heavy tasks. Hence we use the Mirror API, which provides the platform for performing heavy computational tasks on a web server.

2.1 Why Mirror API?

Mirror API is a web based platform for app development on Glass. This makes Mirror API very flexible in terms of development and gives developers freedom in terms for choosing any programming language, framework, and hosting environment [11]. One can develop a Glassware app in any programming language that runs on a web server and can issue web requests, one can even change the programming language as your Glassware evolves.

The Mirror API apps are installed and operate exclusively on a remote server. This makes application development very rapid, since the programmer does not have to load the app onto an emulator or device [11]. Since the Mirror API is web server based, the deployment for the apps is also very quick. Unlike native apps, we do not have to push the

entire code for users to download; instead we only have to save changes on the server and it comes into effect whenever the system syncs automatically [11]. Running the code on a web server also saves battery life on the device as energy is only consumed in sending and receiving data from timeline cards, which is very little [11].

While the Mirror API has a lot of advantages, it also has some pitfalls that we have to work with. The complete workflow of the application depends upon internet connectivity. Firstenberg and Salas [11] observe that Glassware that uses Mirror API operates as a low-latency vehicle, which is to say it distributes its payload to its intended target in the order of a few seconds, but there is going to be some network lag. In other words, even though Glass tries to distribute its load to obtain a quicker response, there is always network lag that affects the output. This makes the complete concept of "real-time" misleading. To tackle this issue of being dependent on the internet, Glass handles offline situations by queuing the sync operations until the network returns [11].

Applications developed on Mirror API are purely event-response driven [11]. This means the Mirror API relies on a user to take explicit actions which trigger events on the Glassware. To add to this passive nature, Mirror API does not have access to full range of sensors of the Glass, for example motion, velocity, acceleration sensors [11].

2.2 Mirror API Architecture

The Mirror API is mainly responsible for manipulating and reacting to changes to the Glass timeline [33]. The Glassware communicates with the Mirror API, which in turn populates the respective user's content of the timeline [33]. All users that use our Glassware have to authorize using Googles OAuth 2.0 in order for the Glassware to manipulate their

timelines. The architecture of Mirror API in Figure 2.1 shows how Glassware and Glass communicate with each other. This architecture is very similar to any standard web application. There is a client machine that communicates with the server through a proxy service. In our case the Google Glass becomes the client machine, Google acts as a proxy, and this proxy service communicates with the server that is running the Glassware. Instead of installing and configuring our own server to host our Glassware, we take advantage of Googles Platform as Service (PaaS) called Google App Engine (GAE)

Figure 2.1 Mirror API architecture [34].

As shown in Figure 2.1, the Glassware communicates with the Google server through the HTTP channel [11]. Glassware uses the standard REST HTTP commands GET and POST to send events to the Glass. Each of these requests consists of an authentication header generated as a result of a successful user authentication using OAuth2 [11]. The communication between Glassware and the Google server is bidirectional - "the Glassware will register a web hook with the proxy, and Google will use HTTP to send events to the Glassware via these interfaces" [11]. Glassware receives data from Google server in one of two forms: JSON object or a URL. All the timeline cards and the textual data that are sent to the Glassware is in JSON format whereas media files are being sent to the Glassware in a URL format. For the media files, instead of sending the whole chunk of data, Google sends the Glassware a URL which it can use to fetch the file [11].

The communication between Google server and Glass headset is labeled as Glass sync. Glass sync phase is responsible for managing all the message passing operations between Google server and Glass headset, such as queueing, optimizing, batching, and handling situations where the device is offline or powered off [11].

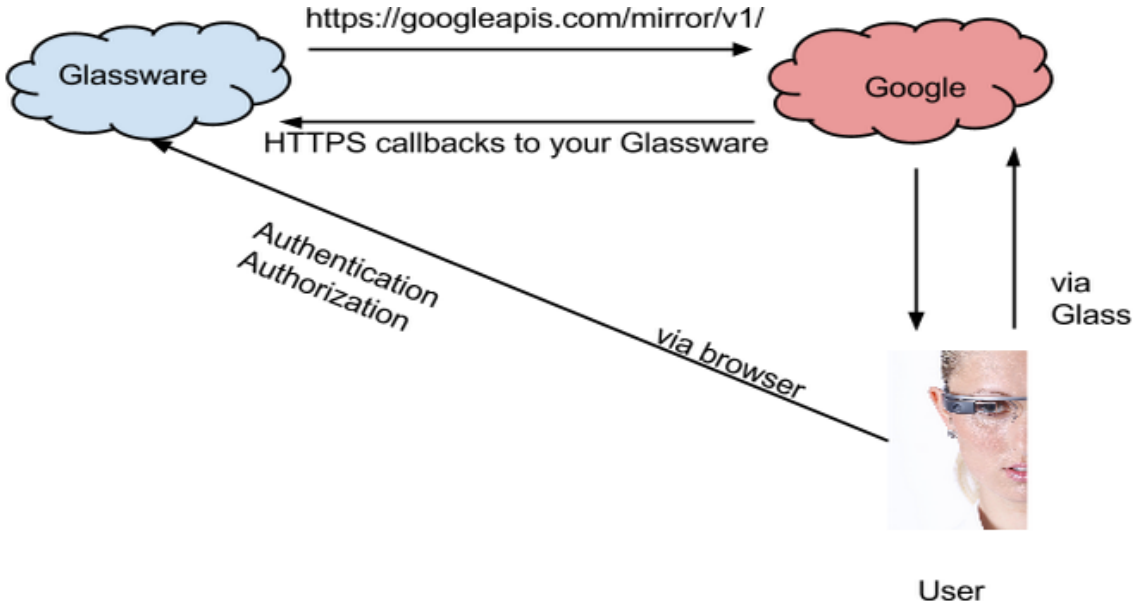


Figure 2.2 Data flow in the Mirror API [11].

Whenever the user invokes a command, shares data with the Glassware, and subscribes to notification from Glassware, these requests are sent to Google. Google servers then forwards these messages in JSON format to the Glassware for processing. This process works both ways: if Glassware needs to send messages to the device, it first sends it to the Google server which delivers it to the intended device and vice versa if the device performs any events, the Glassware is notified via the Google server. From the information flow between the Glassware, the Google server and Glass device, we can conclude that the Glass device or the user never directly communicates with the Glassware.

2.3 OAuth 2.0

Our Glassware can interact with any approved users Glass device through HTTP requests to the Mirror API. Googles server communicates directly with the users Glass device on the behalf of our Glassware. But before the Glassware starts sending out requests to the Glass, we would need authorization for the Glassware to read from or write to the users Glass data [33]. The authorization is not only required to seek permission to manipulate user's timeline but also to indicate that our Glassware application can be flagged as real [33].

Google makes the use of OAuth 2.0, which is a "standard way for network clients to request permission to use a resource on behalf of a user, and for those resources to indicate the client is authorized to take the requested actions" [11], to handle the authorization for the Glass. Although OAuth has grown since 2007, when it was first created by Twitter [35], it still has authorization at its core [11].

API services such as Mirror API cannot trust the client applications or Glassware to behave properly. An unchecked client application can claim to represent a user and then create a disorder that the user of the device never expected. To prevent the misuse of the user's resources, OAuth acts as a check in the process. The Glassware must request permissions to access certain resources or activities, known as scopes, which the users must explicitly approve [11]. To gain approval for resources from the user, OAuth "specifies how the Glassware sends the user over to a site controlled by the Google server where the server can get permissions directly from the user" [11]. OAuth also specifies how the acknowledgment of the permissions are to be sent back to the Glassware.

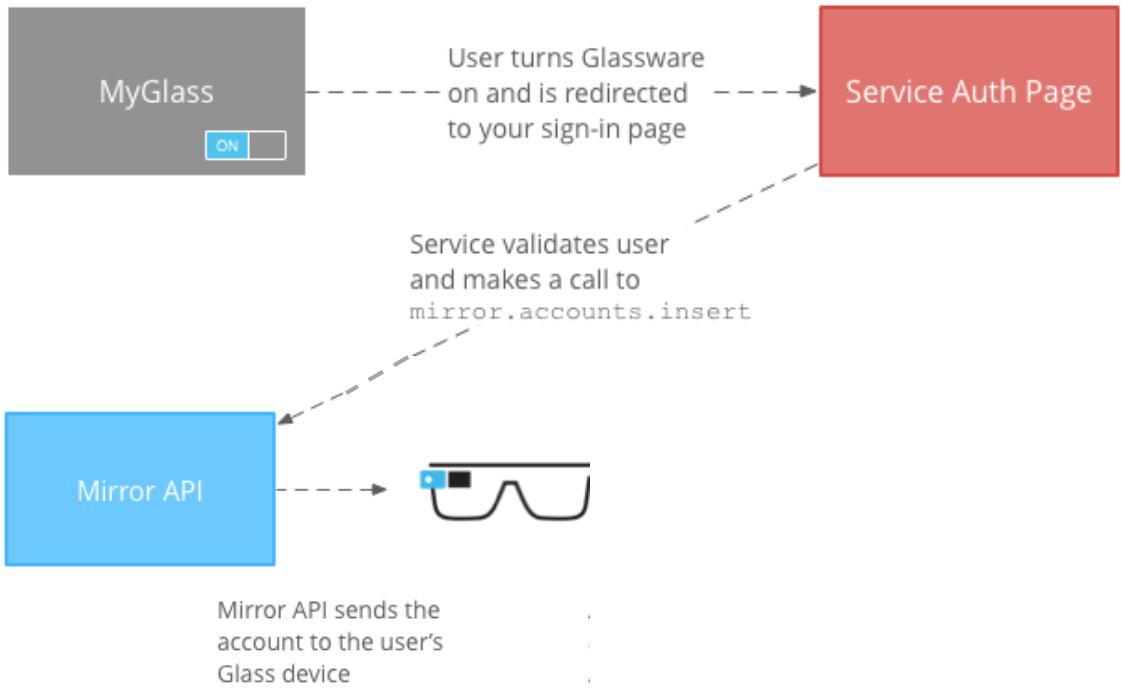


Figure 2.3 Installing and using Glassware [34].

2.3.1 The OAuth dance

To start using our Glassware on their own device, users have to activate the Glassware from the 'MyGlass' website or mobile app. To start the service of our Glassware, the user selects the respective card and clicks the toggle switch "on". This opens up a new window that "starts an elaborate dance" [11] or in other words triggers a series of events, discussed below, between our Glassware and Googles system.

Step 1: Redirect to Google for authorization.

When the new window opens up, on clicking the toggle switch, it is redirected to our website (a server where the Glassware is hosted) for authorization. The next step is for our Glassware server to request that the user authorize our Glassware to access resources. This is achieved by redirecting to Google server for authorization and authentication.

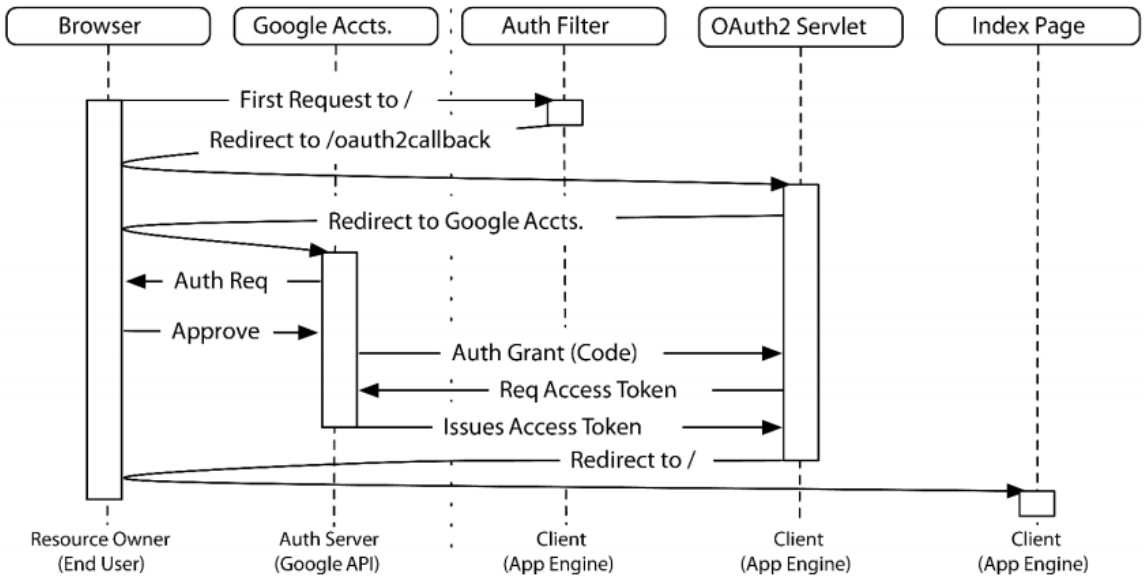


Figure 2.4 OAuth 2.0 protocol flow [33].

Step 2: Authorization and activation.

In this step the user will have to login and authenticate themselves before they can authorize any requests. Once the user successfully logs in to the system, the user is presented with an application approval screen. In this page the user is presented with information about various scopes that our Glassware requested. Each Glassware is assigned a 'client_id' and 'client_secret' when the app is created. The Glassware can use these credentials to identify itself. We send these two parameters in our request to the authorization server, so that the server knows which client is making the consent request.

Step 3: Authorization code.

Once the user approves our request, the Glassware server receives an authorization code. There is no need to store this code, as it is valid only for a few seconds.

Step 4: Exchange for access token.

The Glassware server sends the authorization code to the Googles authorization server. The authorization code is used in exchange for a more long-lived access token and an even longer-lived refresh token.

Step 5: Dance complete.

Once the authorization code is validated by the authorization server it issues a reusable access token and a refresh token. An access token has a time-to-live which helps ensure security. If the token ever fell into the wrong hands, an attacker would be limited in how much time he had to access a user's resources.

2.3.2 Accessing resources

After we receive our access token we have completed all major steps to our dance. We can now make use of this access token to perform all the activities with the users Glass-sending the welcome message, registering contacts, sending updates, etc. The next few steps explain the sequence of events that take place while accessing resources of the device.

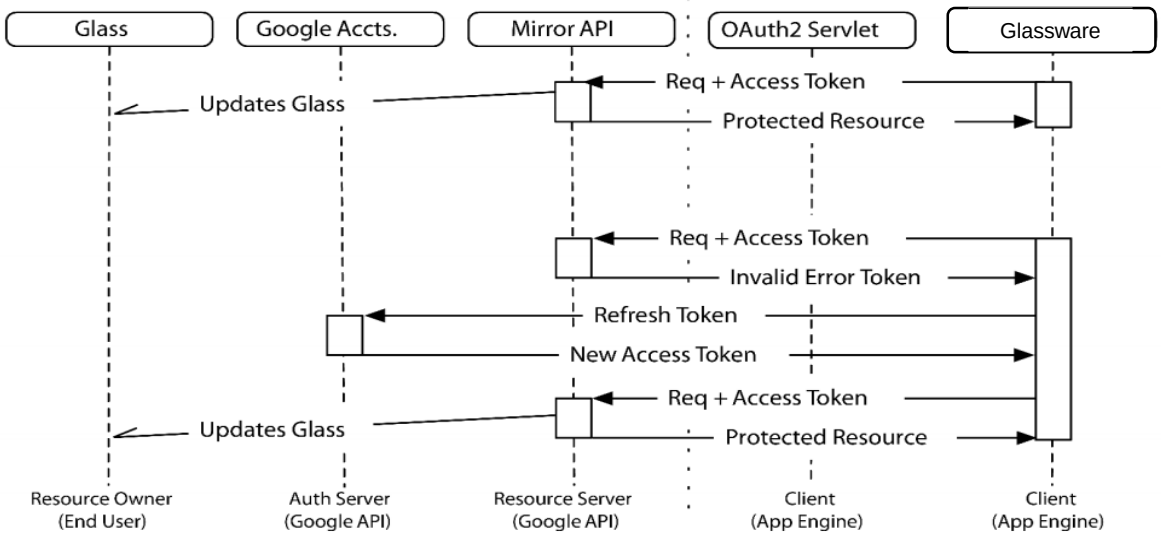


Figure 2.5 Using access token to issue Mirror API request [33].

Step 1: Any request that our Glassware sends to Mirror API must send the access token along.

Step 2: All responses from the Mirror API are protected resources. Googles own system ensures that the access token is authentic and has access to the requested API resources.

Step 3: If the access token expires, a new access token is requested by our Glassware from the authorization server by using the refresh token.

2.4 Workflow of the applications

The workflow is adapted from Googles Mirror API documentation [34] and has the following steps,

1. The user authenticates with OAuth 2.0 and our Glassware stores the credentials.
2. As soon as the OAuth dance gets completed, our Glassware inserts its contact information on the user's device.

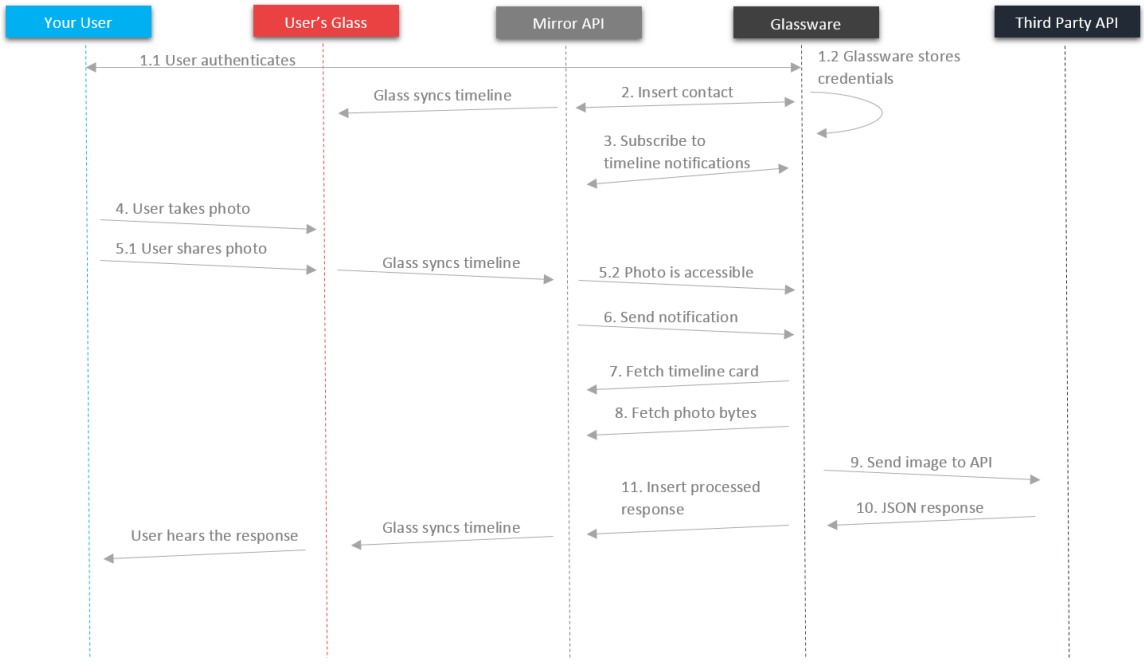
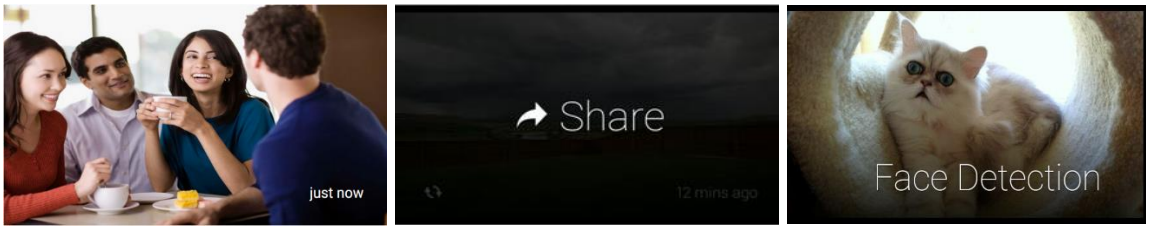


Figure 2.6 Workflow of Glassware [34]

3. Next, our glassware subscribes to the respective users timeline by inserting a timeline subscription for the timeline collection. This completes the setup for the Glassware.
4. To make use of the Glassware, the user takes a picture.
5. The user then shares this picture with the Glassware which gives it access to the timeline card associated with the picture.
6. Our Glassware receives a notification about the picture being shared. This notification links to the timeline item containing the shared picture.
7. Our Glassware examines the notification and fetches the timeline card that contains the photo using the id from the notification.
8. After fetching the timeline card, the Glassware uses the attachment ID to get hold of the picture.
9. This picture is then sent to a third party API for further processing.
10. After the processing is completed, the API returns the response back to the Glassware in JSON format.
11. The Glassware then processes this response and inserts it into a new timeline card. This timeline card is then inserted back into the users timeline with the response fetched from the third party API.

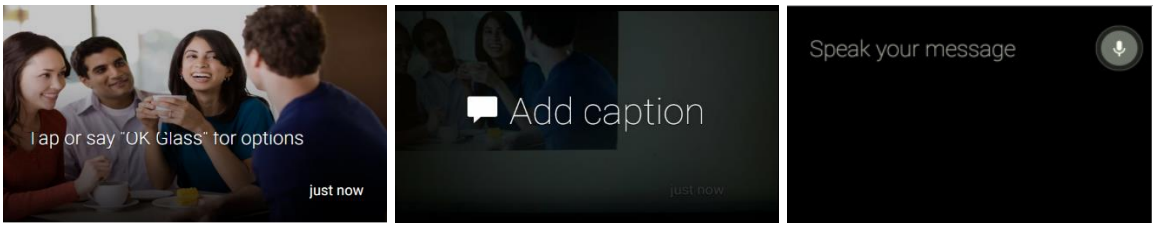
2.5 Application 1: Face Detection

This application was developed to help the people who are visually limited to understand more about the people in their surroundings. They can use the app to identify physical features of people nearby, such as age, gender, emotion, distance from the user, etc.



(a) (b) (c)

Figure 2.7 Sharing captured image. (a) Image captured by the user. (b) "Share" option on Glass. (c) Face Detection Glassware as a contact on users device.



(a) (b) (c)

Figure 2.8 Adding captions. (a) Instructions to access menu. (b) "Add caption" menu option. (c) Speech recognition of device, listening to the user.

The user who is visually challenged takes an image of people around him or her using Glass, as shown in Figure 2.7 (a). The user then gives a voice command to the Glass to "share" this image with our Glassware Face Detection as shown in Figure 2.7 (b) & (c).

We also present the user with additional options before sharing the image with the Glassware, as shown in Figure 2.8 (a). The visually impaired users can access these

additional options menu by either using the voice command “OK Glass” or by tapping the trackpad. On entering the menu the user gets an option to “Add caption” to the image Figure 2.8 (b). This option is used to add the users question to the image. Selecting the “Add caption” option opens up a new card which listens to the users question and performs speech recognition on it, converting it to text.

Once the device completes speech recognition, it attaches the question to the image and shares it with Glassware. This process of sharing actually takes place by creating a copy of the original card and sending the URL of the copy to the Glassware. All the tasks that Glassware performs take place on the copy of the timeline card instead of the original.

The Glassware receives a notification that a timeline card has been shared with it. The app extracts both the image and the question for further processing. It converts the image to Base64 encoded message and sends it to Rekognition API [12] in a POST HTTP request. This request contains a set of parameters such as the API public key, API secret key, etc. The API then examines these parameters and performs operations based on them. On completion of all operations, the API sends back the results in the form of JSON objects.

This JSON response is initially checked for number of faces detected by the API. If the API fails to detect any faces in the image, a timeline card is directly inserted into the users Glass stating the same. If it does detect faces in the image, then the Glassware performs further processing based on the question asked by the user.

The application tries to extract keywords from the users question over the next few steps. The application recognizes six features: age, gender, emotion, sleep, distance, and conversation, and tries to identify keywords that relate to these six features. In particular, we allow the following keywords:

Age: children, adults, and kids.

Gender: male, female, men, women, boys, and girls.

Conversation: chatting, talking, and conversing.

Emotion: happy, angry, sad, calm, and confused.

Sleep: awake and sleep.

Distance: far, near, closest, and farthest.

To extract these keywords from the question, the application breaks the complete question into individual words called tokens. For example, the question "How many happy people can you see?" is broken into 7 individual tokens as follows 'How, 'many' , 'happy', 'people', 'can', 'you', see'.

Next, the application considers all the tokens and identifies the keywords amongst them. To identify if a token is a keyword or not, the application performs a search operation inside a hash map that stores all the features and their respective keywords. For instance, the application will successfully identify 'happy' as a keyword out of the previously mentioned 7 tokens. After identifying all the keywords it stores them and their respective features in a search result hash map. For instance, after the application identifies 'happy' as a keyword, it will store emotion (feature) as the key and happy as its respective value in the search result hash map.

The application uses this search result hash map to filter the JSON response received from the API. For instance if the search result hash map consists of the following key-value pairs, (emotion, happy) and (gender, male), then the application will first filter all results in the JSON response with emotion as happy and send this response for further

filtering based on gender. After all the keys in the search result hash map are used for filtering, we obtain the final JSON response that the user is interested in.

The results obtained after filtering are then sent to a function module that converts the raw JSON object data into a proper English sentence based on keywords (found in the search result hash map) extracted from the question. This sentence is inserted into a timeline card and sent back to the user.

The user can hear a small notification sound when the card sent by the Glassware successfully gets inserted to his or her timeline. Since visually impaired people cannot read the content on the card, a menu option has been added to the card labeled as read aloud. On selecting this option the device will read out the content on the card to the user through the bone conduction transducer audio system.

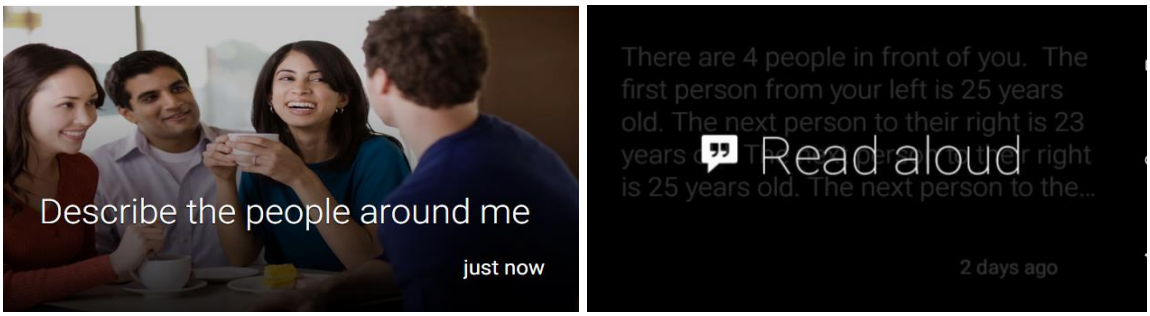


Figure 2.9 Read aloud cards (a) Copy of the original timeline card with question attached to it. (b) “Read aloud” on response card sent by Glassware.

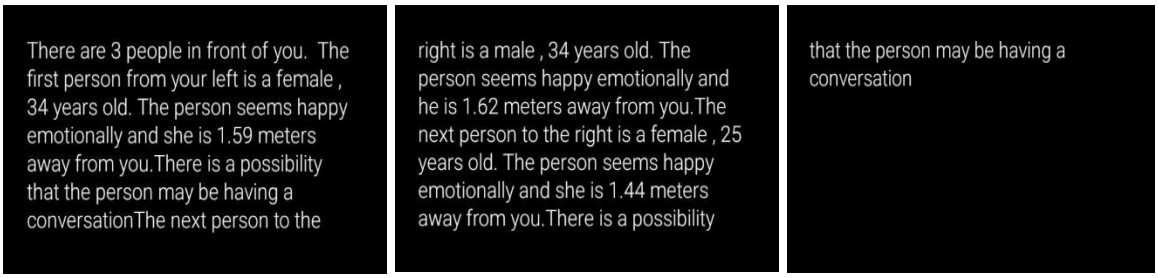


Figure 2.10 Paginated response cards

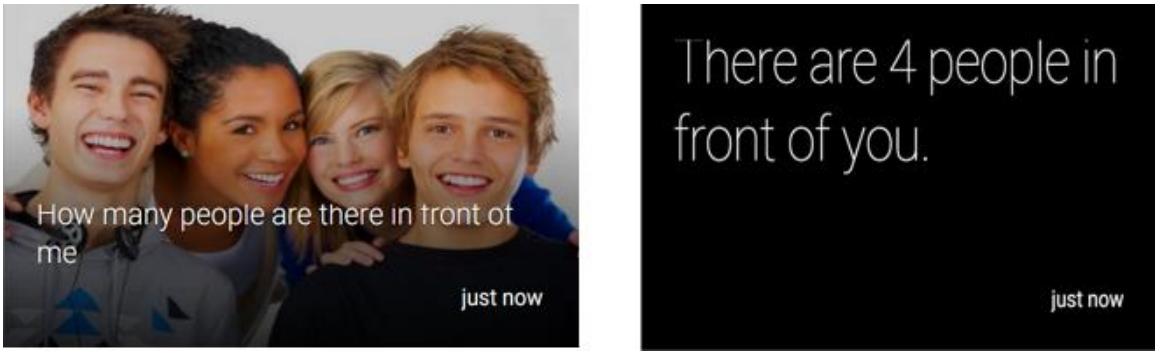


Figure 2.11 Example 1: Counting the number of people in front of the user

Figure 2.12 Example 2: Finding the age of everyone around the user.

Figure 2.13 Sample questions supported by the system.

2.6 Application 2: Object recognition

The second application was developed keeping in mind difficulties faced by visually impaired people in identifying objects used in their day-to-day lives. The application performs recognition of objects based on personalized image tags. The people who are visually challenged can train the application to learn about objects that they have

difficulty in identifying. The application gives the user the ability to tag these objects as per their personal choice, and later it memorizes objects under same name tag.

The user first has to take an image of the object that he or she wants the application to learn. After taking an image of the object, the user shares it with the Glassware. Similar to the first application, during the process of sharing the image of the object with Glassware, the user is given a menu option to "Add caption" to the image. The user can make use of this menu option to add a personal tag to the image. For instance, the user can click a picture of a can of a coke that he or she uses daily, and add a caption which says "tag this object as coke", as shown in Figure 2.14.

Figure 2.14 Cards showing object tagged as coke.

Just like the first app, Glassware receives an exact copy of the original timeline card which has the image along with a caption added to it. The app extracts the tag provided to the object from the caption. It then sends the image converted to Base64 format along with its tag in an HTTP request to the Rekognition [12] API. The API stores this image with its tag in a personal database created for the respective user. On successful storage of the image and tag, the API sends back an acknowledgment to the Glassware which in turn sends a timeline card back to the user with "The object is tagged XYZ" written on it. The user will get a notification on receiving this card and he or she can choose the menu option

to “Read aloud” the content. This card will let the user know that the API has successfully stored the object with its correct tag.

Once the user is done tagging all the items, he will send a train command card to the API, as shown in Figure 2.15. This card will trigger the API to perform deep learning on all the objects and their respective tags. This learning phase ranges from a few seconds for 1000 images [36]. As soon as the API is done training, it informs the Glassware, which sends a timeline card to the user mentioning the completion of training.

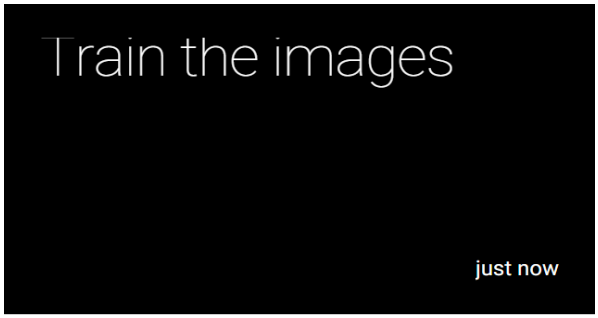


Figure 2.15 The train command card sent to the API.

Now that the API has learnt about all the objects and their respective tags, our app is ready for the user to test. Next, whenever the user faces an issue in identifying an object, he or she will click an image of the object and add the following caption to the image: “search for this object” or “recognize this object”, and send it to Glassware. The Glassware will inspect the card for a caption and accordingly send the image to the API with instructions.

The instruction that Glassware sends out is either to perform search on the image or to perform recognition on the image depending on the users caption. For the search instruction, the API searches the personal image set of the user to find similar objects to the current image [36]. The search instruction is useful in identifying objects that the user has tagged previously. This, for example, is helpful to identify cans of a known brand while

shopping for groceries and can also be helpful in searching for objects in the users kitchen or other places. The recognition instruction recognizes objects in the new image and is useful in scenarios where the user has never come across an object and has never tagged this object before or the user is completely unaware of the object.

Figure 2.16 Response for search instruction.

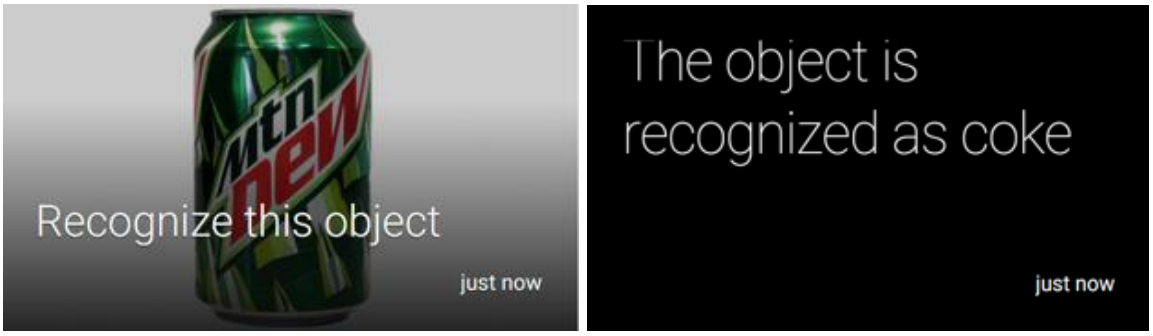


Figure 2.17 Response for recognize instruction.

In both the cases, once API has completed the processing, it will send the result tags to the Glassware in JSON format. The Glassware parses the JSON to understand the tags from the response. It then converts the obtained tag in an English sentence and sends it back to the user by inserting a new card on the timeline.

Chapter 3: Results

The previous chapter highlighted details about how Glassware works and the architecture behind it. In this chapter we discuss results on the performance of the Glassware that we developed.

To test the usability and stability of our applications and also to see how Google Glass performs as an assistive device, we conducted some preliminary evaluations. Our examinations focused on how well our applications perform against Vizwiz [8] in terms of speed and usability. In other words, our goal is to see how well the automated technology performs against current "human powered access technology" [5]. Our applications were tested to see how quick and accurate they respond to a users question. Since the design of our application is such that both our input and output modalities are based on speech, we also evaluate how well speech recognition works on Google Glass.

3.1 Face detection results

3.1.1 Dataset

For our experiments on the Glassware, we created a dataset with two parts. The first part of the dataset included input images. We collected 15 images and added them to the dataset. The input images for our tests were of two types, easy and hard, depending upon the how many people faced the camera. In easy images all the people present in the

image were facing the camera, whereas in hard images not all people present in the image would face the camera. We created this bifurcation to simulate a real world situation where not everyone would be facing the user at all times. Also due to restrictions on privacy we avoided taking images of real people, and instead we took photographs of people on a computer screen or images of printouts with faces from the internet and captured those with the Glass.

The second part of the dataset included input questions. It consists of 105 questions spread equally across 7 categories. The categories are as follows,

Number of people: This category fundamentally counts the number of people detected in a given image. The user can ask questions such as "how many people are present around me?"

Age: Age category informs the users of the age of people around them. It can also classify children and adults. The user can ask questions such as "Are there children present?" or "How many adults do you see?"

Gender: The questions under the gender category informs the users about the gender of people around them. The questions can be as follows "How many females can you detect?", "What is the gender of everyone around me?"

Emotion: The emotions category lets the users know about the emotion of people surrounding them. The category can detect emotions like happy, sad, angry, calm, and confused. Questions under this category are "Are people happy around me?", "How many sad people do you see?" etc.

Sleep: Users can ask questions related to sleep category to know if people around are awake or sleeping. This category has questions such as “Are people around me awake?” or “Is anyone sleeping around me?”

Conversation: The conversation category lets the users know if people around them are having a conversation. This category includes questions such as “Are people chatting to me?” or “Are people around me having a conversation?”

Distance: This last category informs the users about the distance between them and everyone around them. Users can ask questions such as “describe the closest person to me.” or “How far the angry people from me are?”

3.1.2 Experimental protocol

For the experiments, we captured images of people on a computer screen and tested them with questions from all the types of categories except for the distance category. For the distance category questions, we captured images of the faces of people by printing them to average size [43] on paper. For every question, we performed analysis on the data received from the response and also on the time required to process the question.

Since we had labeled input images as easy and hard, it sometimes became difficult to decide if an answer was correct or incorrect. There were situations for the hard images where if the user asks a question like “how many females are present around me?”, the system would accurately answer with respect to people looking towards the camera but fail to detect people not facing the camera. This could mean there might be a female with her back towards the user which the application would not be able to detect. In this case we

cannot state that the response by the application is completely correct or incorrect. Instead we term it as partial, where “partial” indicates partially correct or incorrect.

3.1.3 Accuracy results

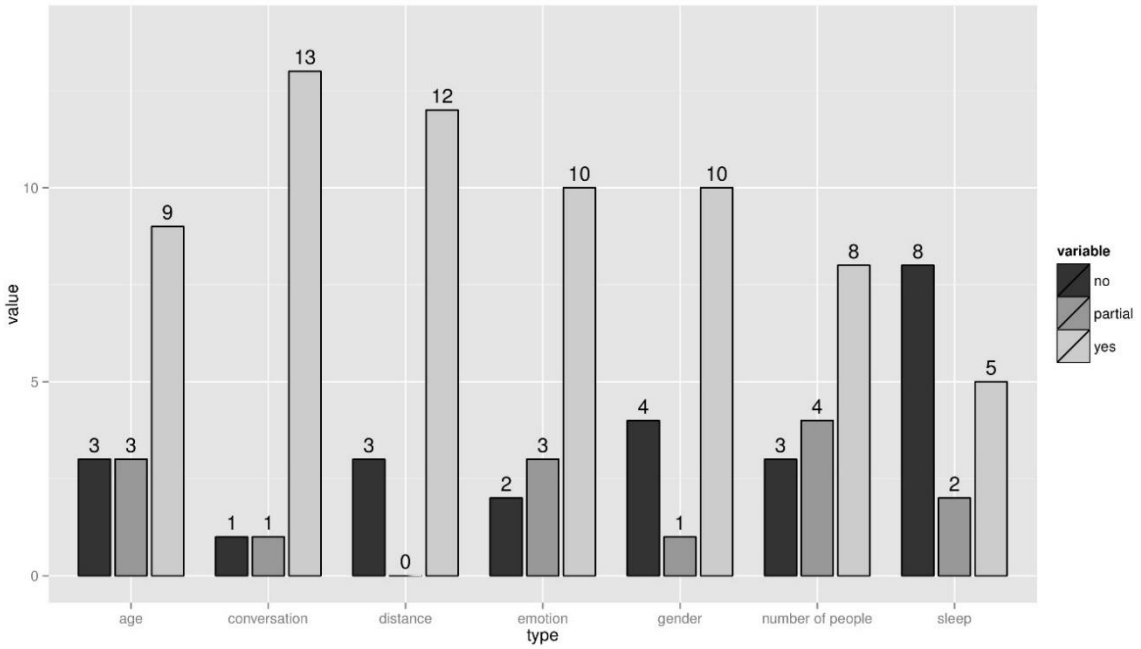


Figure 3.1 Performance for individual categories.

Figure 3.1 indicates the accuracy for each category, on the number of answers which are correct, incorrect or partial in every category. As per Figure 3.1 we observe that the ‘conversation’ category had most of the answers correct whereas the sleep’ category failed in majority of the questions. This mainly happened because 5 out of 8 questions that failed speech recognition belonged to the sleep category. This also indicates that the sleep’ category may need some improvements in the application. The remaining categories performed well since they had mostly correct answers.

For every question, we observed two things: firstly, if it was correctly recognized by the speech recognition module of the device and secondly, if the question was

answerable or not with respect to the image. Figure 3.2 statistics show that we got only 8 out of 105 questions incorrectly recognized by speech recognition software of the device and only 7 questions out of 105 were unanswerable with respect to the image.

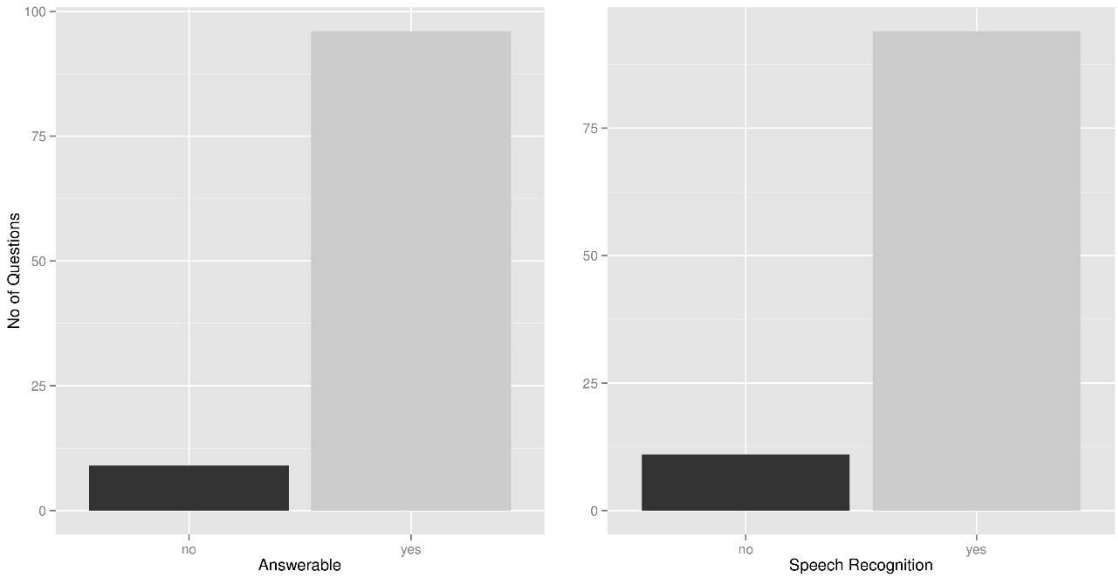


Figure 3.2 Evaluation of questions.

3.1.4 Timings results

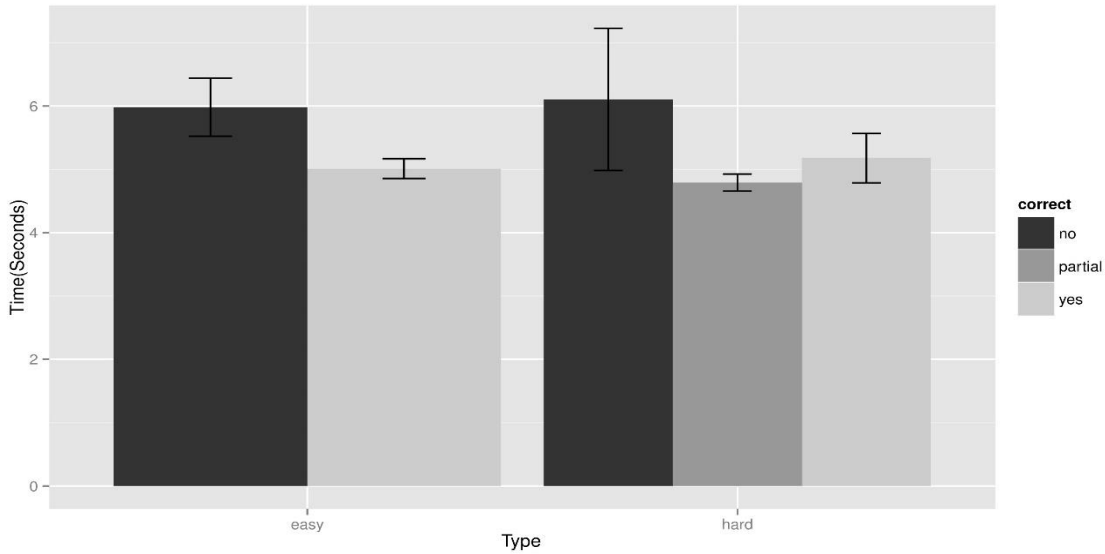


Figure 3.3 Response time for application one.

From Figure 3.3 we can observe that the tests where input images were easy had, for all questions, an average response time of 5.17 seconds and for tests where input images were hard had, for all questions, the average response time of 5.25 seconds. One can conclude that overall the average response time across any type of input image is 5.21 seconds. The average response time reported by Bigham et al. [6] for VizWiz [8] is 133.3 seconds for the first answer received across all questions. Our application is thus 25 times faster than VizWiz [8] on average. This makes our application closer to real-time than VizWiz [8].

The graph in Figure 3.3 also indicates that the response time of the application is almost same time for both easy as well as hard type of images. We can observe that the response time is higher when the application answers incorrectly. This is mainly because it checks for all possibilities before returning the result, which means it takes more time due to more processing.

Type of image	Targeted features	Questions	Response time (seconds)	Correct answer	Correct speech recognition
easy	number of people	how many people can you see	5.92	yes	yes
easy	age	age of everyone around me	5.71	no	yes
easy	emotion	emotion of everyone around me	4.44	yes	yes
easy	sleep	is anyone sleeping around me	5.92	yes	yes
easy	conversation	are people having conversation around me	5.72	no	no
easy	gender	how many females can you see	4.41	no	yes
easy	gender	what is gender of everyone around me	6.94	no	yes
easy	number of people	how many people can you see	10.44	no	yes
easy	age	what is age of everyone around me	4.29	no	yes
easy	emotion	what is emotion of everyone around me	5.62	yes	yes

easy	sleep	how many people are awake around me	7.26	yes	no
easy	conversation	how many are having a conversation around me	4.09	yes	yes
easy	gender	how many females can you see	4.43	yes	yes
easy	gender	what is gender of everyone around me	4.44	yes	yes
easy	age	what is age of everyone around me	4.51	no	yes
easy	number of people	how many people can you see	6.1	yes	yes
easy	emotion	what is the emotion of everyone around me	4.97	no	yes
easy	sleep	are people awake around me	4.52	no	no
easy	sleep	are people awake around me	6.23	no	no
easy	conversation	are people having conversation around me	5.1	yes	yes
easy	conversation	forgot to add caption	4.62	yes	yes
easy	gender	what is the gender of everyone around me	4.54	yes	yes
hard	number of people	how many people can you see	4.7	no	yes
hard	age	what is age of everyone around me	4.3	partial	yes
hard	emotion	what is emotion of everyone around me	4.72	partial	yes
hard	sleep	how many people are asleep around me	4.64	partial	yes
hard	conversation	are people having a conversation around me	4.92	partial	yes
hard	gender	what is the gender of everyone around me	4.5	no	yes
hard	distance	is the closest person to me angry	3.8	yes	yes
easy	distance	what is the emotion of closest person to me	7.5	no	yes
easy	distance	what is the emotion of closest person to me	6.4	no	yes
easy	distance	describe closest person to me	4.12	yes	yes
easy	distance	describe closest person to me	5.92	yes	yes
hard	number of people	how many people can you see	5.6	yes	yes
hard	age	how many adults can you see	4.9	yes	no
hard	gender	what is gender of everyone around me	4.11	yes	yes
hard	emotion	how many happy people can you see	3.82	yes	yes
hard	sleep	are people awake in front of me	4.4	no	no
hard	conversation	how many people are talking	3.97	yes	yes
hard	number of people	how many people can you see	5.44	partial	yes
hard	age	what is the age of everyone around me	5.72	yes	yes
hard	gender	what is the gender of everyone around me	3.88	partial	yes
hard	sleep	are people sleeping near me	4.35	no	yes

hard	conversation	are people talking to me	5.08	yes	yes
hard	emotion	what is the emotion of people around me	5.17	partial	yes
hard	distance	is closest person to me angry	4.13	yes	yes
hard	distance	describe closest person to me	5.56	yes	yes
hard	gender	how many females can you see	4.54	yes	yes
easy	number of people	how many people can y	6.88	yes	yes
easy	age	how many adults can you see	4.43	yes	yes
easy	gender	how many females can you see	4.42	yes	yes
easy	sleep	are people awake near me	4.08	yes	yes
easy	conversation	how many people are talking near me	5.59	yes	yes
easy	emotion	how many happy people can you see	5.13	yes	yes
easy	distance	how far are people from me	4.26	yes	yes
easy	number of people	how many people can you see	5.39	yes	yes
easy	age	how many adults can you see	4.36	yes	yes
easy	age	Are there any kids near me	4.08	yes	yes
easy	sleep	are people asleep near me	4.04	yes	yes
easy	conversation	is anyone talking near me	4.5	yes	yes
easy	emotion	what is emotion of everyone around me	4.39	yes	yes
hard	number of people	how many people can you see	5	partial	yes
hard	age	how many kids can you see	4.81	partial	yes
hard	gender	how many boys can you see	4.26	yes	yes
hard	sleep	are people awake	5.13	partial	yes
hard	conversation	how many people are talking	4.72	yes	yes
hard	emotion	how many sad people can you see	4.43	no	no
hard	distance	how far are the kids from me	5.66	yes	yes
hard	number of people	how many people can you see	6.5	yes	yes
hard	age	are there kids around me	5.46	yes	yes
hard	gender	are there girls around me	4.57	partial	yes
hard	sleep	are people sleeping near me	4.14	no	no
hard	conversation	is anyone talking with me	4.47	yes	yes
hard	emotion	are there sad people near me	4.45	yes	yes
hard	distance	how far are happy people from me	4.32	yes	yes
hard	number of people	how many people can you see	5.44	partial	yes

hard	age	how many kids can you see	5.72	yes	yes
hard	gender	what is the gender of everyone around me	3.88	partial	yes
hard	sleep	are people sleeping near me	4.35	no	yes
hard	conversation	are people talking to me	5.08	yes	yes
hard	emotion	what is the emotion of people around me	5.17	partial	yes
hard	distance	is closest person to me angry	4.13	yes	yes
easy	number of people	how many people can you see	7.17	yes	yes
easy	age	are adults present around me	4.29	yes	yes
easy	emotion	what is emotion of everyone around me	5.62	yes	yes
easy	sleep	how many people are awake around me	7.26	yes	yes
easy	conversation	how many are having a conversation around me	4.09	yes	yes
easy	emotion	how many happy people can you see	4.43	yes	yes
easy	number of people	how many people can you see	5.56	yes	yes
easy	age	how many kids can you see	3.93	yes	yes
easy	gender	how many females can you see	4.62	no	yes
easy	sleep	is anyone sleeping around me	7.48	no	no
easy	conversation	are people talking to me	4.1	yes	yes
easy	emotion	are people angry with me	4.38	yes	yes
easy	distance	describe closest person to me	4.72	yes	yes
hard	number of people	how many people can you see	4.58	no	yes
hard	age	how many kids can you see	3.44	yes	yes
hard	gender	how many males can you see	14.14	yes	no
hard	sleep	is anyone sleeping near me	13.5	no	yes
hard	conversation	are people chatting near me	4.44	yes	no
hard	emotion	do you see angry people	7.82	yes	yes
hard	distance	am I in a safe area	12.11	no	yes
easy	distance	how far are angry people from me	5.56	yes	yes
easy	distance	describe closest person to me	3.93	yes	yes

Table 3.1 Results of face detection.

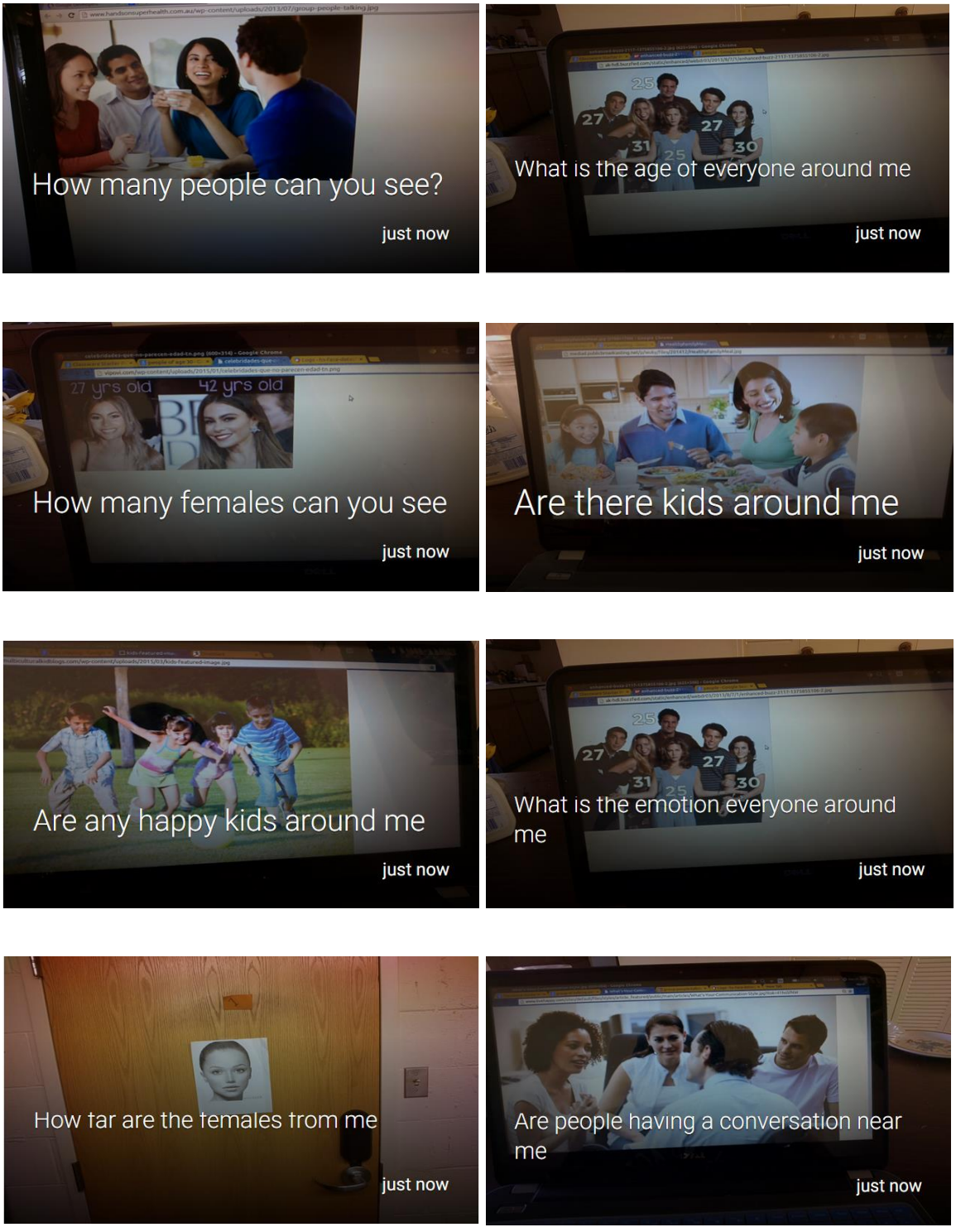


Figure 3.4 Sample of dataset for face detection Glassware.

3.2 Object recognition results

3.2.1 Dataset

For the dataset of our second Glassware, object recognition, we decided to choose 10 objects that are used in daily lives and tag them accordingly. These objects were as follows: plates, can of beans, can of coke, can of mountain dew, apple, bananas, orange, box of salt, plate, headphones, and coffee mug. The dataset had 50 images in total, with 5 images for each object (each from a random angle). The dataset was divided into a training set of 30 random images and a test set of the remaining 20 images.

3.2.2 Experimental protocol

Our object recognition Glassware was developed to help the people who are visually impaired to identify objects used in daily lives. We conducted an experiment to test the reliability of our application, in which we tested the application on 20 images from the test set, with either search or recognize instruction chosen randomly, to gather data for analysis.

3.2.3 Accuracy Results

The Glassware recognized 12 out of 20 objects from the test set, which indicates an accuracy of 60%. Although the performance of the Glassware in terms of accuracy is not

very high but fortunately the Glassware also determines a confidence score for every result. This confidence score can be used to decide if the answer is likely to be correct or not.

To understand the reliability of the confidence score, we plot a graph which is similar to Precision vs Recall graph, with number of questions answered correctly (representing precision) vs number of questions asked (representing recall). This graph, as shown in Figure 3.5, indicates the chances of getting a correct answer from the application. This indicates that the users can use this graph to configure glassware as per their preference of precision and recall. For instance, users can have a configuration where the Glass can answer 100% of their questions but only be right 60% of the time. Alternately the users can configure it to be very conservative: it might answer only 30% of questions while saying "I don't know" to the rest 70 %, but would be right 100% of the time for those 30% answers.

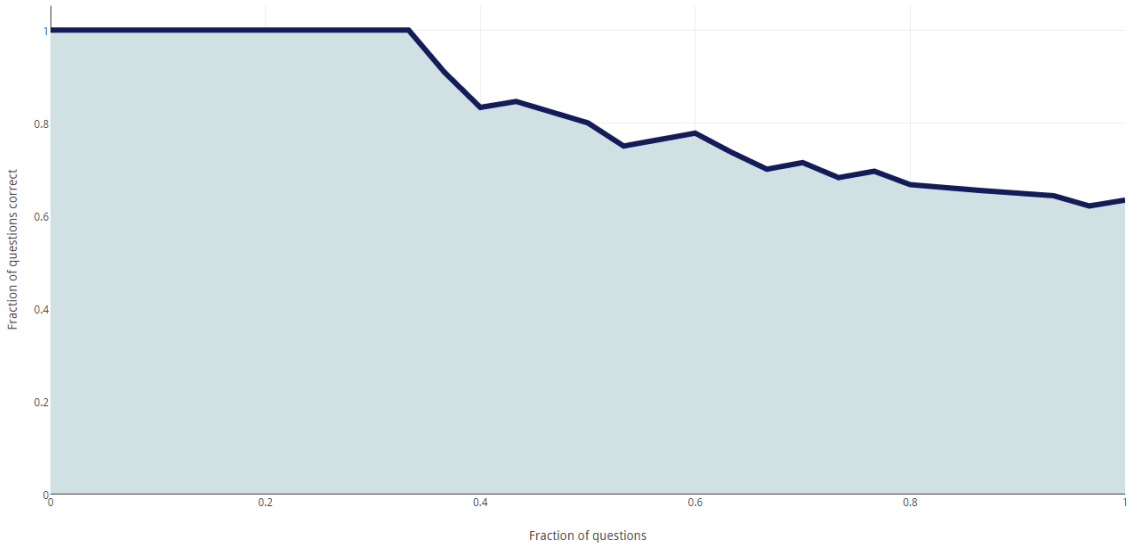


Figure 3.5 Fraction of questions answered correct vs fraction answered.

We also decided to evaluate which instruction performs better, search or recognize. We selected 10 random images from the test set and provided them to the Glassware to perform both search and recognize on each of these images. On comparing the results it was observed that the first response for both instructions matched exactly not only for identifying the object but also for the confidence score. For instance, both correctly recognized a test image as a can of coke having 89 percent confidence score. This suggests that the user can use any instruction in any situation and still get the same result.

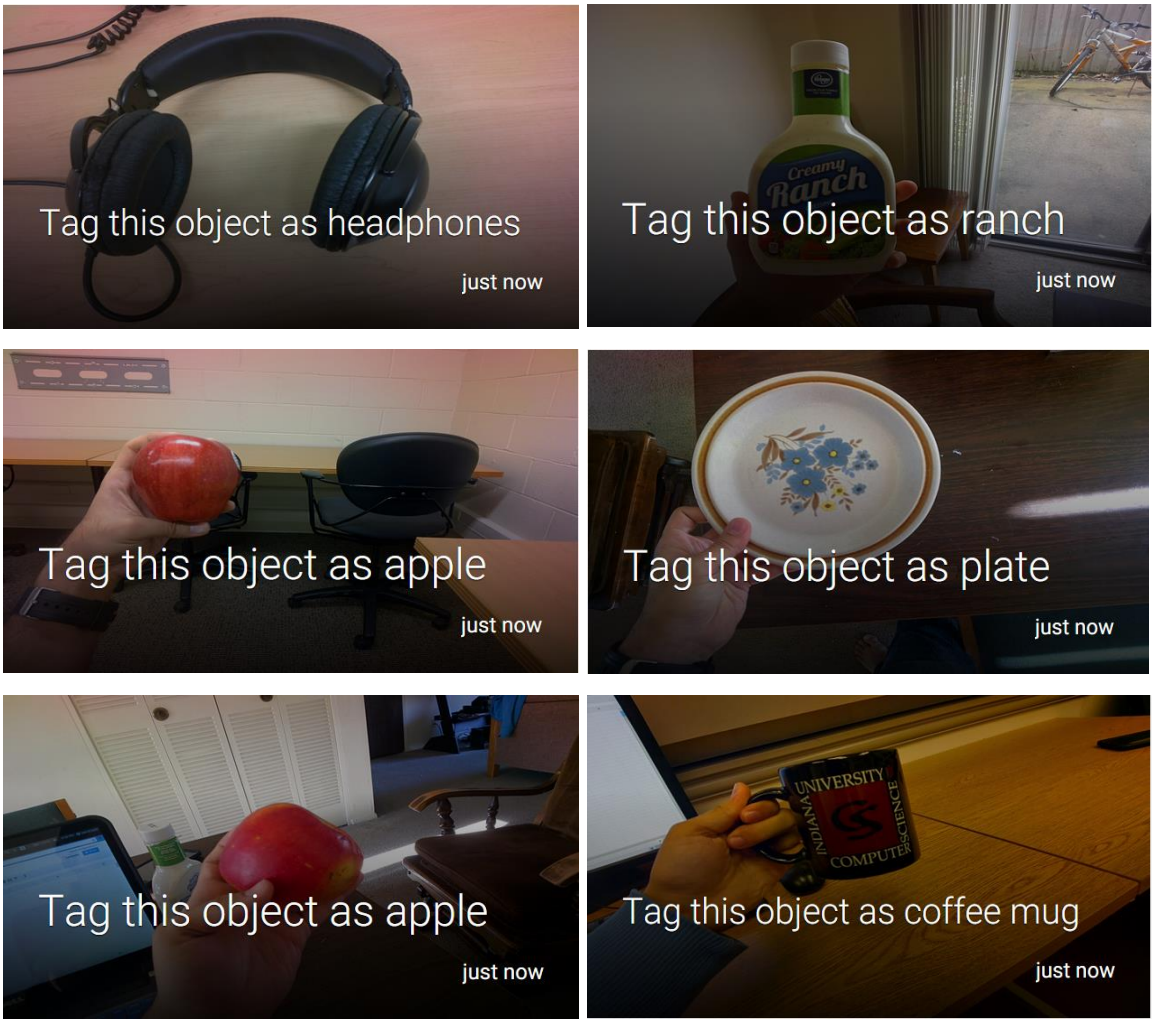


Figure 3.6 Sample of dataset for object recognition Glassware.

True label	Identified label	Confidence score	Instruction
Apple	Apple	93	3.16
Oranges	Oranges	91	2.79
Beans	Beans	89	3.67
Beans	Beans	89	2.88
Apple	Apple	89	4.11
Apple	Apple	89	3.18
Oranges	Beans	88	1.75
Coke	Apple	78	1.95
Coke	Coke	75	3.29
Beans	Box of salt	59	3.57
Box of salt	Box of salt	59	3.4
Box of salt	Beans	54	2.9
Plate	Plate	53	3.16
Plate	Plate	53	2.54
Headphones	Coffee mug	40	4.52
Headphones	Coffee mug	36	2.09
Bananas	Bananas	33	3.08
Oranges	Apple	26	2.67
Headphones	Headphones	12	4.52
Beans	Bananas	10	2.78

Table 3.2 Results of object recognition.

3.2.4 Timing results

After testing the Glassware on the test set, it was observed that the average response time for the search instruction was 3.24 seconds and the average response time for recognize instruction was 3.1 seconds. One can conclude that the overall average of the response time for the Glassware is 3.17 seconds. Even though the accuracy of our Glassware is probably not high enough for real world use, the average response time is 42 times faster than VizWizs [8]. This signifies that the application could respond in near "real-time" to the questions asked by the users.

3.3 Discussion

On 31st May 2013 Google put an abrupt end to any speculation about it supporting facial recognition on Glass. The Project Glass team, in a Google+ post [37] stated that the team has noted concerns and opinions of people around the possibilities of facial recognition on Glass. In the post Google said "we won't add facial recognition features to our products without having strong privacy protections in place. With that in mind, we won't be approving any facial recognition Glassware at this time." [37].

Due to the above restriction imposed by Google we used third party APIs to perform face detection for our first application. Also to ensure that we do not breach privacy of anyone, we decided to use images of people on a computer screen to test our applications instead of capturing real people.

The Glass was designed as a "perpetually-on device" [11], which meant that the battery life for the device was estimated to last one day of typical use. Unfortunately, the camera based services such as video recording or taking many images drastically drains

the battery. The Glass then displays the following message to warn the user ‘Glass must cool down to run smoothly’. Due to the low battery life the visually impaired users will have to use external battery packages or power banks such as Gazerg [44] to be able make the use of applications throughout the day.

The other hardware issue of the device is with the bone conduction transducer audio system. Although the system helps to protect privacy against bystanders, it is likely not audible enough to the user themselves in a crowded and noisy environment. People who are visually limited can only make use of the applications in a less noisy environment where they can hear the output loud enough. The other solution to tackle the audio issue would be to connect Glass wirelessly with an external haptic feedback device which conveys the on screen content to the user.

The tests performed on our applications show that they respond to the users query in near real-time, but both our applications fail in-case the user does not have internet connection on the device. For situations where network connection drops, the glass sync operation will queue all the messages and send them to the device as soon as it connects back. This indicates that user will have to face delays in such situations.

The idea behind developing these apps was to build a proof of concept for the applications. To get things working we had to confine our applications to work in specific scenarios. The face detection application currently only works in a constrained environment where it can give answers only for a limited number of features related to people around the user, and does not support generic questions to the applications. There needs a lot of work to be done in this area, which will allow the users to ask generic questions.

Chapter 4: Conclusion

In this chapter we summarize our work in the thesis and discuss the future possibilities of Google Glass and our Glasswares as assistive technology.

4.1 Summary

We have presented two applications for Google Glass, enabling a user to ask questions about people around him or her, and to identify objects based on their personal tags. Although our apps run in a constrained environment, they are significantly faster than apps that use "human powered access technology" [5] approaches. Our applications can act as new prototypes for automated answering apps and in general for assistive apps for Google Glass. Our work has opened up a new horizon of assisting the people who are visually impaired using Google Glass, which is waiting to be explored

4.2 Future Work

Through our work we tried demonstrate new types of automated answering apps using Google Glass and computer vision. We addressed two issues frequently faced by people who are visually impaired: knowing about people present around them, and helping

them identify objects. Although our face detection application focuses on answering questions confined to a limited set of features, future work of our application we could use, more sophisticated techniques, like graphical models or a bot similar to 'Google Now' [40] that can answer general questions. The object recognition application could also be expanded to perform recognition on all objects and not just personalized tag objects.

Our applications show that Google Glass can be very useful as an assistive device. Our next step would be to develop more apps on Glass that can be used by people who are visually impaired. There could be an application which performs optical character recognition of text in the image and reads it out loud to the user. Another idea could be related to navigation. Google Glass can be reliable in terms of navigating on road, but there could be an app which helps the users who are visually impaired to navigate indoors too.

People who are visually impaired often have issues with input modalities of a device. To explore new input modalities we could try to combine glass with other wearable technologies such Mind wave, an electroencephalogram (EEG) headset [42] or Myo, a gesture control wristband [43]. The Mind wave [42] allows the user to give commands using signals from his or her brain, which can be more convenient to people who are visually challenged than operating phones with the hand or talking to Glass. The Myo [43], a wristband with gesture controls, could be incorporated with Glass to operate using certain gestures, so the device could be operated discreetly.

References

[1] Wearable computing. http://www.inc.com/magazine/201312/ryan-underwood/the-wearable-computing-boom.html

[2] Wearable disabilities. http://fortune.com/2015/02/10/wearables-disability/

[3] Brady, E., Morris, M. R., Zhong, Y., White, S., & Bigham, J. P. (2013, April). Visual challenges in the everyday lives of blind people. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 2117-2126). ACM.

[4] Velázquez, R. (2010). Wearable assistive devices for the blind. In Wearable and autonomous biomedical devices and systems for smart environment (pp. 331-349). Springer Berlin Heidelberg.

[5] Bigham, J. P., Ladner, R. E., & Borodin, Y. (2011, October). The design of human-powered access technology. In The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility (pp. 3-10). ACM.

[6] Bigham,J.P., Jayant, C., Ji, H., Little, G., Miller, A., Miller, R.C., Miller, R., Tatarowicz, A., White, B., White, S. and Yeh, T. (2010). VizWiz: nearly real-time answers to visual questions. In Proceedings of the ACM Symposium on User Interface Software and Technology. p333-342.

[7] LookTel. http://www.looktel.com/

[8] VizWiz. http://www.vizwiz.org/

[9] Brady, E., Morris, M. R., Zhong, Y., White, S., & Bigham, J. P. (2013, April). Visual challenges in the everyday lives of blind people. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 2117-2126). ACM.

[10] Ahmed, T., Hoyle, R., Connelly, K., Crandall, D., & Kapadia, A. (2015, April). Privacy Concerns and Behaviors of People with Visual Impairments. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (pp. 3523-3532). ACM.

[11] Firstenberg, A., Salas, J. Designing & Developing for Google Glass. (December 2014). Published by O' Reily Media, Inc.

[12] Rekognition. www.rekognition.com

[13] Alchemy API. www.alchemyapi.com

[14] Starner, T. E. (1999). Wearable computing and contextual awareness (Doctoral dissertation, Massachusetts Institute of Technology).

[15] WHO. http://www.who.int/mediacentre/factsheets/fs282/en/

[16] Hersh, M., & Johnson, M. A. (2010). Assistive technology for visually impaired and blind people. Published by Springer Science & Business Media.

[17] Ye, H., Malu, M., Oh, U., & Findlater, L. (2014, April). Current and future mobile and wearable device use by people with visual impairments. In Proceedings of the 32nd annual ACM conference on Human factors in computing systems (pp. 3123-3132). ACM.

[18] Jayant, C., Acuario, C., Johnson, W., Hollier, J., & Ladner, R. (2010, October). V-braille: haptic braille perception using a touch-screen and vibration on mobile phones.

In Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility (pp. 295-296). ACM.

[19] Manduchi, R., & Coughlan, J. (2008). Portable and mobile systems in assistive technology (pp. 1078-1080). Springer Berlin Heidelberg.

[20] Schauerte, B., Martinez, M., Constantinescu, A., & Stiefelhagen, R. (2012). An assistive vision system for the blind that helps find lost things (pp. 566-572). Springer Berlin Heidelberg.

[21] Kane, S. K., Jayant, C., Wobbrock, J. O., & Ladner, R. E. (2009). Freedom to roam: mobile device opportunities and challenges for people with visual and motor impairments. In Proceedings of ACM SIGACCESS Conference on Computers and Accessibility, ACM Press, 115-122.

[22] Dawe, M. (2007, October). Understanding mobile phone requirements for young adults with cognitive disabilities. In Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility (pp. 179-186). ACM.

[23] Kane, S. K., Bigham, J. P., & Wobbrock, J. O. (2008, October). Slide rule: making mobile touch screens accessible to blind people using multi-touch interaction techniques. In Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility (pp. 73-80). ACM.

[24] Kurniawan, S. (2008). Older people and mobile phones: A multi-method investigation. International Journal of Human-Computer Studies, 66(12), 889-901.

[25] Watanabe, T., Miyagi, M., Minatani, K., & Nagaoka, H. (2008). A survey on the use of mobile phones by visually impaired persons in Japan (pp. 1081-1084). Springer Berlin Heidelberg.

[26] Borodin, Y., Bigham, J. P., Dausch, G., & Ramakrishnan, I. V. (2010, April). More than meets the eye: a survey of screen-reader browsing strategies. In Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A) (p. 13). ACM.

[27] Starner, T. (1997). The Cyborgs are Coming or The Real Personal Computers. Unpublished paper submitted to Wired.

[28] Rhodes, B. J. (1997). The wearable remembrance agent: A system for augmented memory. Personal Technologies, 1(4), 218-224.

[29] Wikipedia. http://en.wikipedia.org/wiki/Wearable_computer

[30] Pasquero, J., Stobbe, S. J., & Stonehouse, N. (2011, May). A haptic wristwatch for eyes-free interactions. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 3257-3266). ACM.

[31] Dakopoulos, D., & Bourbakis, N. G. (2010). Wearable obstacle avoidance electronic travel aids for blind: a survey. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 40(1), 25-35.

[32] Ha, K., Chen, Z., Hu, W., Richter, W., Pillai, P., & Satyanarayanan, M. (2014, June). Towards wearable cognitive assistance. In Proceedings of the 12th annual international conference on Mobile systems, applications, and services (pp. 68-81). ACM.

[33] Redmond, E. Programming Google Glass. (2013). Published by The Pragmatic Programmers, LLC.

[34] Google Mirror API. https://developers.google.com/glass/develop/mirror/index

[35] OAuth Wikipedia. http://en.wikipedia.org/wiki/OAuth#History

[36] Rekognition developer guide. https://rekognition.com/developer/object

[37] Googles post. https://plus.google.com/u/0/+ProjectGlass/posts/fAe5vo4ZEcE

[38] Lambda Labs. https://lambdal.com/

[39] Facial recognition. https://facialrecognition.com

[40] Google Now. https://www.google.com/landing/now/

[41] Mind wave. http://store.neurosky.com/products/mindwave-1

[42] Myo. https://www.thalmic.com/en/myo/

[43] Wikipedia Human Head. http://en.wikipedia.org/wiki/Human_head

[44] Gazerg. http://gazerg.com/